Data Mining on the Sisyphus Dataset: Evaluation and Integration of Results

Thomas Gärtner[1,2], Shaomin Wu[1], and Peter A. Flach[1]

[1] Department of Computer Science, University of Bristol,
Woodland Road, Bristol BS8 1UB, U.K.
{gaertner,shaomin,flach}@cs.bris.ac.uk
[2] Knowledge discovery team, AiS, GMD,
Schloß Birlinghoven, D-53754 Sankt Augustin, Germany
thomas.gaertner@gmd.de

Abstract. This paper describes our work on the Sisyphus challenge dataset, which includes both classification and clustering tasks.. Key aspects of the work are the evaluation and integration of multiple models by means of ROC analysis. We indicate a simple method of forcing classifiers to cover the whole of the ROC space. In conclusion, we outline several promising research directions.

1 Introduction

The Sisyphus dataset [1] is an excerpt from a data warehouse which is employed to collect and analyze data from the private life insurance business at Swiss Life. Swiss Life expects to find concept descriptions for classes of the data, and to find clusters of instances in the data. A concept description for a class is an abstract representation of a subset of the data that can be used to predict the class of an unseen instance. A cluster is a set of data instances that are very similar to instances within the same cluster, but very different to instances of a different cluster.

The Cross-Industrial Standard Process for Data Mining (CRISP-DM) [2] distinguishes six different data mining phases, among which the main technical phases are data preprocessing and modelling, while the other phases mostly concern management issues. Within the SolEuNet project the Sisyphus dataset was used as a testbed for remote collaboration in Data Mining tasks. The main focus was on the two technical phases, as the other phases are within the responsibility of the project management committee and mostly performed locally. The requirements (for tools supporting collaborative data mining) found by this experiment will be described elsewhere [3].

This workshop paper will focus on the data modelling techniques applied by one of the groups participating in the Sisyphus experiment. This group, consisting of the three authors of this paper, focused on applying propositional learning algorithms to the data and comparing the results using ROC analysis [4]. Most of the applied algorithms are state-of-the-art, as implemented in the WEKA toolkit [5], however, also some less well-known algorithms were applied. The final classifier is a hybrid

classifier made up by the convex hull of all classifiers in ROC space. The so-called ROC convex hull classifier is known to perform at least as good as any of the classifiers it is constructed from.

The outline of the paper is as follows. In Section 2 we sketch the background of this work regarding the Sisyphus data mining tasks, ROC analysis, and data preprocessing. Section 3 presents our results for one of the classification tasks, and Section 4 addresses one of the clustering tasks. Section 5 concludes with some possible future research directions.

2 Background

In the following subsections we briefly sketch the data mining tasks that are part of the Sisyphus challenge, ROC analysis, propositionalisation of the originally multi-relational data, and removal of the 'not applicable' cases.

2.1 The Sisyphus Tasks

The Sisyphus data mining problem consisted of three different learning tasks, two supervised and one unsupervised. The first supervised learning task consisted of finding a concept description for a subset of the 17267 partners involved in insurance contracts. Those partners are marked with class=1 in the relation 'taskA'. The other task consisted of finding a concept description for a subset of the 12945 households the partners live in. Those households are marked with class=1 in the relation 'taskB'. The unsupervised learning task is to find a clustering of households that takes the insurance policies in which they are involved into account.

As the approaches in both classification tasks are more or less similar, this paper focuses only on classification of partners and on the clustering of households.

2.2 ROC Analysis and the Convex Hull Classifier

ROC analysis [4] is usually applied if some conditions of the classification, e.g., the misclassification costs or the distribution of the test data, are unknown or subject to change. In ROC space, each classifier is represented by two coordinates, corresponding to its true-positive (TP) rate and its false-positive (FP) rate. The TP-rate is defined as 'number of positive instances correctly classified' divided by 'total number of positive instances'; the FP-rate is analogously defined as 'number of negative instances incorrectly classified' divided by 'total number of negative instances'.

By splitting the commonly used predictive accuracy into two coordinates, the behaviour of each classifier can later be analysed with respect to actual conditions. Such 'actual conditions' are described in ROC space by a so-called iso-performance line. For instance, in a model that takes misclassification costs into account the iso-performance lines are straight lines with a slope determined by misclassification costs and class distributions. The diagonal connecting (0,0) and (1,1) is such an iso-performance line, indicating minimum obtainable performance: each point on this line can be obtained without training.

The ROC convex hull method allows to create a hybrid classifier that is at least as good as any single classifier under any condition. This classifier is described by a set of locally optimal classifiers, i.e., classifiers that are optimal for a certain slope range of iso-performance lines.

2.3 Propositionalisation

The original dataset provided by Swiss Life was multi-relational. To apply propositional learning algorithms these relations had to be joined into a single table such that each individual is described by a single row of the table. This kind of joining relations into a single table is known as propositionalisation and is not a trivial task. Propositionalisation can usually not be done automatically and requires sufficient understanding of the data to handcraft the set of attributes of the table that is to be created. As indicated above this paper describes only a part of a bigger experiment on collaborative data mining. The propositionalisation of the data in this experiment was performed by Andrea Lüthje from Dialogis. The dataset used in the classification and clustering tasks described below are based on this propositionalised data.

2.4 Removing 'not-applicable' Cases

A number of partners and households were classified as 'not applicable'. From the data description it is not clear what the semantics of these 'not applicable' cases is. The task was to find a concept description for the Class=1 instances, and the task description did not say anything about the other Classes. Therefore the supervised learning tasks could be accomplished in three different ways: firstly, the 'not applicable' cases could be removed; secondly, they could be merged with the Class=2 cases; and finally, the Classification could be used as is, i.e., with three different Classes.

For several reasons we will consider in this paper only the problem of distinguishing two classes:

- Several of the learning algorithms we used are restricted to binary classification.
- Especially in the Partner domain concept descriptions of the 'not applicable' cases were very easy to induce, and almost every multi-class learning algorithm found a 'top-level rule' for identifying those cases.
- Model assessment using ROC analysis is more convenient when dealing with two-class problems.

3 Classifying Partners

In this section we present our results for one of the classification tasks. Sections 3.1 and 3.2 give details of the data and of the selected modelling techniques, respectively. Section 3.3 gives a ROC analysis of the results, while Section 3.4 gives additional results in an attempt to cover the full ROC space.

3.1 Data Understanding and Preprocessing

There are 17267 partners distributed as follows:

Table 1. Class-distribution of the partner relation

Class	Partners	Percentage	Percentage of applicable partners
Not Applicable – 0	3945	22.8%	
Yes – 1	10723	62.1%	80.5%
No – 2	2599	15.1%	19.5%

As described in the section above the 'not applicable' cases have been removed and thus the prior distribution is roughly 4:1.

3.2 Selected Modelling Techniques

The following learning algorithms have been applied to the data: OneR, Naïve Bayes, J48, linear SVM, and WBC_{SVM} (all but the last taken from the Weka toolkit). These algorithms will briefly be introduced in the remainder of this section. If not stated differently below, the reader is referred to [5] for a more detailed description.

OneR is a very simple learning scheme. It uses a classifier model based on a single attribute to classify the data. Each nominal value or interval of numeric values is assigned a class corresponding to the most frequent class given that attribute value in the training data. To reduce the danger of overfitting the minimal size of each 'bucket', i.e., the minimal number of instances in an interval, has been set to 6 (default). The attribute is chosen such that the accuracy on the training data is highest. The class predicted for test instances is determined by the value of the selected attribute. The models inferred by OneR consist of a set of rules that all test the same attribute.

Naïve Bayes is a classifier that induces a very simple statistical model of the instances of each class. It is based on the idea of predicting the most likely (maximum- posterior) class, and thus on the Bayes-optimal classifier. However, as calculating this maximum- posterior class reliably requires huge amounts of data and is computationally expensive, some simplifying assumptions about the instance space are made. In particular, it is assumed that the attributes describing each instance are conditionally independent given the class. Although this assumption is known to be violated very often, Naïve Bayes is still a good classifier in many cases.

J48 is Weka's implementation of the well-known decision tree learner C4.5 (Revision 8). It can generate unpruned trees and trees pruned with reduced error pruning or with subtree raising. For a detailed description of C4.5 the reader is referred to [6].

Support vector machines [7,8] or SVMs use linear classifiers to implement non-linear class boundaries. This is achieved by using a non-linear mapping to transform the input space into a representation space which is usually (but not necessarily) of much higher dimension. Learning in the new representation space is performed by looking for the maximum margin hyperplane. Two important aspects of SVMs are

that the search for this hyperplane can be performed by solving a quadratic optimisation problem, and that the feature space has only to be known implicitly, by means of a kernel function. In the case of *linear* SVMs the kernel corresponds to the inner product in the original representation space. Contrary to nonlinear support vector machines, linear SVMs do increase the domain understanding as the weight vector can be calculated explicitly and more weight is given to more discriminating features.

WBC_{SVM} [9] (Weighted Bayesian Classification based on Support Vector Machines) is based on the idea of improving naïve Bayes classification by assigning a different weight to each conditional probability. The algorithm is similar to the support vector machine approach, but uses a special kernel function. In contrast to other well known kernel functions, the one used in WBC_{SVM} depends on instance and class distributions. The weights defining the separating hyperplane have a direct interpretation as feature weights in a weighted naïve Bayes classifier.

3.3 Model Assessment using ROC Analysis

Because of the size of the dataset and the large number of attributes, performing crossvalidation was not feasible. Instead, the dataset was split in a training set containing 70% of the instances and a test set containing 30% of the instances. The data was split randomly, and only constrained by the desire to have similar class distributions in both sets.

Each learning algorithm was used to induce a model of the training data. This model was then in turn used to predict the class of the test instances. True-positive and false-positive rates have been documented and are shown in the table below:

Table 2. Results of different algorithms in ROC space

Classifier	FP-rate	TP-rate	Accuracy
AllNeg	0.0	0.0	0.195
J48	0.334	0.952	0.897
Naïve Bayes	0.445	0.881	0.818
Linear SVM	0.465	0.986	0.899
WBC_{SVM}	0.548	0.991	0.800
OneR	0.691	0.959	0.834
AllPos	1.0	1.0	0.805

AllNeg and AllPos are default classifiers that always predict one of the given classes. They are included in the table for reasons of completeness.

From the results in the above table it can be seen that it is easy to achieve good classifications on the majority (positive) class, but difficult to find a good description of the minority class. However, under certain circumstances (very different distribution of the test set, or high misclassification costs) it is very important to find good descriptions of the minority class, i.e., to try to cover the full ROC space.

It should also be noted that while another classifier trained by other data mining experts performed better than the algorithms described above, it still did not succeed in achieving very low FP-rates. In particular, TILDE [10] is a relational learning algorithm that achieved a FP-rate of 0.257, a TP-rate of 0.995, and an accuracy of

0.947. Given these rates, the convex hull classifier made up by AllPos, AllNeg, and TILDE covers all other classifiers in ROC space. This is a surprisingly strong result, especially as a FP-rate of 0.257 is still very high. The following section will introduce a variant of naïve bayes and describe how this variant can be used to obtain models that cover the full ROC space.

3.4 Covering the full ROC space

The naïve bayes classifier outlined above can be biased to predict one class more often than the other by changing the prior probabilities. This can either be done by training naïve bayes on a resampled training set with different class distributions, or by simply biasing the class prior probability distribution. The second approach was followed in these experiments.

With different biases imposed, naïve bayes achieved the following classifications:

Table 3. Results of the biased naïve bayes algorithm in ROC space

Classifier	FP-rate	TP-rate	Accuracy
Naïve Bayes	0.445	0.881	0.818
Biased Naïve Bayes	0.196	0.751	0.761
Biased Naïve Bayes	0.163	0.713	0.737
Biased Naïve Bayes	0.117	0.658	0.702
Biased Naïve Bayes	0.081	0.599	0.660

As it turns out, none of the biased naïve bayes results is covered by any of the classifers described before. Thus the attempt to cover a bigger part of the ROC space by imposing an artificial bias on naïve bayes was successful.

Looking at the convex hull of the classifiers, however, only one of the biased naïve bayes classifiers is not covered by the convex hull. The following table describes the final convex hull classifier by showing the locally best classifiers and the slope range (of iso-performance lines) for which they are optimal.

Table 4. The ROC convex hull classifier

Locally optimal Classifier	Slope Range
AllPos	[0.000, 0.007]
TILDE	[0.007, 2.250]
Biased Naïve Bayes	[2.250, 6.739]
AllNeg	[6.739, Inf]

Please note again, that – under any condition – this convex hull classifier is at least as good as any of the classifers described above.

4 Clustering Households

We proceed by describing our work on the clustering task. Sections 4.1 and 4.2 give details of the data and of the selected modelling techniques, respectively. Section 4.3 analyses the experimental results, while Section 4.4 analyses in more detail one of the clusters that appears most meaningful.

4.1 Data Understanding and Preprocessing

The household dataset consists of 12934 distinct households, distributed as follows:

Table 5. Class-distribution of the household relation

Class	Partners	Percentage	Percentage of applicable cases
Not Applicable - 0	5605	43.3%	
Yes – 1	3705	28.6%	50.5%
No – 2	3624	28.0%	49.4%

As described previously the 'not applicable' cases have been removed and thus the prior distribution is roughly 1:1.

4.2 Selected Modelling and Assessment Technique

Two conceptually very different clustering techniques are described in the literature: feature and instance clustering [11]. Feature clustering is a technique for finding clusters of similar features, while instance clustering aims to find clusters of similar instances. Here, we describe our experiments with instance clustering. Several instance clustering approaches are proposed in the literature, for instance, fuzzy clustering [12], self-organising maps (SOM) [13], average linkage [14] and machine learning approaches such as COBWEB [15].

Among these, we have tried SOM (implemented in Clementine [16]), COBWEB and EM (implemented in Weka [5]). However, SOM ran out of memory on our machines, and COBWEB seems very sensitive to twp parameters: the acuity and the cutoff [5]. So, in this paper we consider only the EM algorithm as implemented in Weka. We are aware that one of the shortcomings of the EM algorithm (and some other clustering algorithms) is their sensitivity to the order of the training instances, and to the selection of initial parameters of the algorithm.

Another general problem with clustering algorithms is that it is usually very difficult to assess the clustering quality. One frequently used assessment of clustering quality is to use a classified dataset, train the clustering algorithm without the class attribute, and finally compare the clusters to the classes. The major shortcoming of this approach is that the clustering algorithm might get bad results even if it found an interesting and important structure in the data. This will happen if the most apparent structure of the data does not correspond to the class structure. Being aware of this problem we will still follow this assessment approach, as it is the one most frequently used in literature. In detail our assessment method works as follows:

Lots of machine learning approaches are prone to overfitor underfit. To avoid both overfitting and underfitting, EM clusters a training set and evaluates the clustering result using a test set. The quality of the clusters found is measured by the log-likelihood, i.e., the logarithm of the likelihood. The bigger this quantity the better the model fits the data. Because of the big dataset and the large number of attributes, performing crossvalidation was not feasible. Instead, the dataset was split in a training set containing 70% of the instances and a test set containing 30% of the instances. The data was split randomly, and only constrained by the desire to have similar class distributions in both datasets.

The main focus of clustering approaches is on the similarity within a cluster and dissimilarity between clusters. We will discuss both similarity and dissimilarity below.

4.3 Assessment of Experimental Results

Two test instances in the household data were found to cause error messages during clustering (error message array contains NaN – can't normalize). For that reason these instances were removed from the test dataset. There are 159 features in the propositionalised household dataset used for clustering (all but the class attribute). The number of clusters to be found was chosen differently for each experiment, it varied between 2 and 13. It was empirically found that the log-likelihood measure is biggest if the number of clusters is set to 8. The results of clustering the household data into 8 clusters is given below:

Table 6. Output of the EM algorithm with number of cluster set to 8.

training data:		**test data:**	
Cluster	Instances	Cluster	Instances
0	148 (3%)	0	85 (4%)
1	103 (2%)	1	50 (2%)
2	175 (3%)	2	72 (3%)
3	4111 (81%)	3	1778 (80%)
4	32 (1%)	4	28 (1%)
5	358 (7%)	5	163 (7%)
6	97 (2%)	6	42 (2%)
7	29 (1%)	7	14 (1%)
Log likelihood: -140.42308		Log likelihood: -146.91822	

The two tables above show the number and percentage of instances in each cluster for both the training and the test dataset. Notice that for all but one cluster the percentage of instances in this cluster is less than 10%. This may indicate that this clustering is not particularly meaningful, in spite of the high log-likelihood.

We will now look in detail into a result that has worse log-likelihood, but where the clusters are more uniformly distributed. The results of clustering the household data into 3 clusters is given below:

Table 7. Output of the EM algorithm with number of clusters set to 3

```
training data:                    test data:
Cluster Instances                 Cluster Instances
       0      1305 ( 26%)               0       609 ( 27%)
       1      2058 ( 41%)               1       866 ( 39%)
       2      1690 ( 33%)               2       757 ( 34%)
Log likelihood: -413.58677        Log likelihood: -411.27219
```

Although the log-likelihood in this experiment is not as good as the one when the number of clusters is eight, the percentage of every cluster ranges from 27% to 39%. This result will be investigated in more detail below.

4.4 Further Investigations

The table below shows the means and the standard deviations of the features regarding insurance policies in each cluster for the test dataset. We see that both means and standard deviations are minimal for all features in cluster 2, intermediate in cluster 1, and maximal in cluster 0. That is, the insurance policies in cluster 2 are closely similar, while those in cluster 0 display more variance.

Table 8. Means and standard deviations of some selected attributes for each cluster

Attribute	Cluster 0	Cluster 1	Cluster 2
PRTYP-9	4.72 ± 2.59	2.48 ± 1	1.74 ± 0.73
PRTYP-10	0.17 ± 1.06	0 ± 0	0 ± 0
PRTYP-11	4.92 ± 2.7	2.49 ± 1	1.75 ± 0.74
PRTYP-12	4.81 ± 2.65	2.46 ± 1	1.74 ± 0.73
PRTYP-17	0.19 ± 0.65	0.05 ± 0.25	0 ± 0.06

Comparing the structure of the clustering to the class structure we get the following result:

Table 9. Comparison of class and cluster structure

	Cluster 0	Cluster 1	Cluster 2	Total
Class 1	403	518	215	1136
Class 2	206	348	542	1196
Total	609	866	757	2232

From the table above we find that 66% of the instances in cluster 0 are in class 1 while only 28% of instances in Cluster 2 are classified as 1. That is, most instances in cluster 0 belong to class 1 while most instances in cluster 2 belong to class 2. One can compare cluster 0 with cluster 2 if one is interested in what the differences between class 1 and class 2 are.

5 Conclusions and Possible Future Directions

This final section summarises our results and points out possibilities for further work. We distinguish between general conclusions, future work on the classification task, and future work on the clustering task.

5.1 General Conclusions and Suggestions

In our work we have tried to follow the CRISP-DM methodology, but we have concentrated on the modelling phase. We have investigated ways of combining multiple models, possibly obtained from different data mining experts, through the ROC convex hull method. Whe have also applied several modelling techniques (classification and clustering) to the same dataset. Aspects of this work which we feel have been particularly successful include: the use of 'forced classifiers' (in this case, biased naïve bayes) to cover the full ROC space; and collaboration between different data mining experts on different sites (in this respect we would also like to mention the work of Andrea Lüthje from Dialogis GmbH, who did the propositionalisation).

Even though different modelling techniques have been applied, this has not been done in an integrated way and this constitutes the first possibility for future work. For instance, one could use the resuls of (feature) clustering as an input to classification.

5.2 Suggestions of Future Work on Classifying Partners in Insurance Policies

Several promising approaches have not yet been tried. Applying feature transformations such as principal component analysis (before the actual classification) could increase the performance. Analyzing the new feature space would in turn increase the domain understanding, as it might reveal dependencies in the data.

Classification accuracy might further be increased by performing parameter optimization on the training data (with cross-validation). As the original representation of the data is relational it is also very promising to apply relational (ILP) algorithms, i.e., algorithms that do not rely on propositionalised data.

Apparently, forcing a classifier (in our case naïve bayes) to cover the whole ROC space was successful. A similar revision is easily implemented in other algorithms such as nearest neighbour, support vector machines, and WBC_{SVM}. Such revised algorithms could lead to major improvements of the final convex hull classifier.

5.3 Suggestions of Future Work on Clustering the Households

Clustering techniques can also be used to find outliers. For example, when the number of clusters is 10, cluster 4 has only one instance. In this case, the instance in cluster 4 is probably an outlier which can be removed. Outlier-checking has two positive aspects: on the one hand, searching for outliers can increase our understanding of the data; on the other hand, the quality of data mining (for example, clustering) will be better after outliers are removed.

Some standardization should be done before clustering since features which have large variances impact on clustering more than those which have small variances.

Standard machine learning packages such as Weka or Clementine mostly implement instance clustering algorithms and not feature clustering techniques. Future work may investigate the structure of the feature space using feature clustering techniques to reduce the dimensionality of the representation. Using fewer features will also speed up other learning algorithms.

Some clustering algorithms are sensitive to the input order of instances. Ensemble methods used in other machine learning techniques (for example, bagging and boosting) may be applied to clustering.

Acknowledgements

This work is supported by the Esprit V project (IST-1999-11495) *Data Mining and Decision Support for Business Competitiveness: Solomon Virtual Enterprise*. Thanks are due to our partners in the project, in particular Andrea Lüthje for pre-processing the data. Thanks also to Joerg-Uwe Kietz of Swiss Life for providing the data. We would also like to thank the two anonymous reviewers for their comments and suggestions.

References

[1] Kietz, J-U, and Staudt, M., KDD-Sisyphus I, PKDD'99, http://research.swisslife.ch/kdd-sisyphus/

[2] Chapman, P., Clinton, J. Kerber, R., Khabaza, T., Reinartz, T., Shearer, C., and Wirth, R., *CRISP-DM 1.0: Step-by-step data mining guide*. CRISP-DM consortium, 2000

[3] Voss, A., Gärtner, T., and Moyle, S., Zeno for Rapid Collaboration in Data mining Projects. *IDDM workshop*, 2001.

[4] Provost, F. and Fawcett, T., Robust Classification for Imprecise Environments. *Machine Learing*, 42(3), pp. 203-131, 2001.

[5] Witten, I.H., and Frank E., *Data Mining - Practical Machine Learning Tools and Techniques with Java Implementations*. Morgan Kaufmann, 2000.

[6] Quinlan, J.R. *C4.5: Programs for Machine Learning*. Morgan Kaufmann Publishers, San Mateo, CA., 1993.

[7] Boser, B.E., Guyon, I. M., and Vapnik, V.N., A training algorithm for optimal margin classifiers. In D. Haussler (Ed.), *Proceedings of the fifth Annul ACM Workshop on Computational Learning Theory* (pp. 144-152). ACM Press, 1992.

[8] Cristianini, N., and Shawe-Taylor, J., *An introduction to Support Vector Machines and other kernel-based methods*. Cambridge University Press, 2000.

[9] Gärtner, T. and Flach, P.A., WBC_{SVM}: Weighted Bayesian Classification based on Support Vector Machines. *Proceedings of the 18th International Conference on Machine Learning (ICML-2001)*, pp. 154-161, 2001.

[10] Blockeel, H., and De Raedt, L., Top-down induction of first-order logical decision trees. *Artificial Intelligence* 101(1-2):285--297, 1998.

[11] D'Agostino, R.B., Dukes, K.A., Massaro, J. M., and Zhang, Z., Data/variable reduction by principal components, battery reduction and variable clustering. *Proceedings of the Fifth Annual Northeast SAS Users Group Conference*, 1992.
[12] Bezdek, J.C., and Hathaway, R.J., Optimization of Fuzzy Clustering Criteria using Genetic Algorithms, *Proc. First IEEE Conference on Evolutionary Computation*, Vol 2, pp. 589-594, 1994.
[13] Kohonen, T., *Self-organisation and Associative Memory*. Berlin: Springer Verlag, 1989.
[14] Johnson, R.A., and Wichern, D.W., *Applied Multivariate Statistical Analysis, 3rd edition*, Prentice Hall, Englewood Cliffs, N.J., 1992.
[15] Fisher, D.H., Knowledge acquisition via incremental concept clustering, *Machine Learning* 2, pp.139-172, 1987.
[16] Clemetine data mining package, SPSS, http://www.spss.com/clementine/.