Towards Statistical Planning for Marketing Strategies

Qiang Yang
Department of Computer Science
Hong Kong University of Science and Technology
Clearwater Bay, Kowloon, Hong Kong, China
qyang@cs.ust.hk

ABSTRACT
AI planning has traditionally dealt with developing sophisticated plans for achieving well-defined goals in well-defined domains. In this paper we consider the problem of building marketing plans for massive customers to achieve a company's financial goal in a business-planning domain. Corporations and institutions are often interested in strategic planning for marketing strategies to target their customers and outperform their competitors. For example, a stockbroker company may draft a marketing plan for retaining valuable customers or for switching a potential customer to a true customer. Planning in these applications consists of market segmentation, marketing-action selection and validation. For such problems, the traditional planning frameworks no longer apply. Instead, planning is done based on statistical reasoning of previous cases and patterns. In this paper, we present a novel framework that incorporates data mining, case based reasoning and planning to support marketing-strategy planning. In our approach, we discover case bases by data mining on the customer database and formulate plans based on the mined cases or "role-models". These plans are not guaranteed to work for each individual; however, based on previous experience, they have a high probability of succeeding. We explore the tradeoff among time, space and quality of computation in this framework. We demonstrate the effectiveness of the methods through empirical results.

1. Introduction
AI planning has traditionally focused on generating plans for a single user to achieve some well-defined goals. In this paper, we consider the problem of generating marketing plans to be acted on massive customers who achieve a company's financial goals, where the actions and goals are only implicitly defined. Our work is motivated from the realistic problems of developing marketing strategies to increase a company's overall profit, and is based on a statistical consideration of the given dataset by building actionable classification models using data mining algorithms. Compared to traditional data mining tasks, we take one step further than simple classification of data; we not only use statistical models to classify customers, but also produce marketing plans to be acted on the customers that make them to switch classes. As we shall see, the traditional planning framework where goals and actions are formally and logically specified no long applies. Instead, we propose a statistical planning framework for constructing these plans using data mining and case based reasoning.

Consider the example shown in Table 1. Suppose that we are given a customer database from a mobile-phone company. The last attribute records whether the customers signed on to a new service contract after some marketing actions, such as sending a free gift, has been applied to the customer. Based on these marketing data, we are interested in knowing what marketing plans would be the most effective in order to increase the chance of valuable customers to sign on to a new service contract. For example, for a customer Basil, we are interested in knowing whether we should give the customer a fee reduction, sending him a new gift or allowing him longer free airtime. We are also interested in how much fee reductions we should provide for Basil to achieve our purpose while keeping the marketing cost at a minimum.

Table 1. A Cell Phone Company's Marketing Planning Problem

		Mobile Phone Data			
		Fee Reduction ($)	Gift	Free calls	Stayed?
Cellphone Users	John	10	Y	80 min	Yes
	Beatrice	20	Y	100 min	Yes
	Mary	30	N	120 min	Yes
	Mathew	20	N	100 min	No
	Steve	15	Y	150 min	No
	Basil	?	?	?	Yes

This example introduced a number of interesting issues for planning. Traditionally, this type of problems were not considered as planning problems, because there are no logically formulated actions with preconditions and effects, no logically provably correct goals and initial states. However, on a close examination, there are many aspects of the marketing plan problem that are of interest to planning researchers. First, although goals are not explicitly given, they can be discovered. For example, a possible goal is to identify a potential role model, say John, as a potential marketing goal for Basil, and formulate marketing actions to make Basil resemble John as much as possible. This may involve giving Basil a fee reduction of $10.00, sending him a gift and allowing for no less than 80 min of free airtime. Second, although there are no logically formulated actions such as the Strips representation, the actions are clearly present. For example, fee reduction for a customer is indeed a potential marketing action that can be taken. Third, similar to considerations in planning with uncertainty, the marketing plans are not guaranteed to work for any particular customer. **B**oth the costs and probability of success are taken into account. The goal here is to maximize the expected utility of the overall marketing plan for all customers.

The marketing plan problem also has a wide range of applications that are beyond business marketing. For example, it can be formulated as an advice generation for students who apply for graduate schools. Instead of rejecting a graduate school applicant with only a "no" answer, we suggest steps that might be taken by the applicant in the future to increase his/her chance of being admitted the next time around.

Table 2. An example customer database.

Customer	Salary	Cars	Mortgage	Loan Approved?
John	50K	2	None	Y
Mary	40K	1	300K	Y
…	…	…	…	…
Steve	40K	1	None	N

Table 3. Prescribed plans for Steve.

Advice for Steve	Salary	Cars	Mortgage
Plan 1	40K→50K	1→2	
Plan 2			0→300K

Similarly, plans can be constructed to give advice to customers who fall short of loan applications. As an example, consider a customer database shown in Table 2. Suppose that we are interested in providing an advice for Steve (the last row) who failed to apply for a bank loan. Obviously, there are many candidate actions that one can advise Steve to take in order to succeed in his next loan application. For Steve, we can advise him to find another job with a salary close to 80K and increase his car number from one to three; this will make him look more like John. Alternatively, we can advise Steve to take up a mortgage from the bank worth at least 300K. This will make Steve look more like Mary. In either situation, Steve might have a higher chance of succeeding than before, but the actions come with different costs. The prescribed actions for Steve are shown in Table 3.

The above-formulated problem can be stated as a combination of data mining problem and case-based reasoning problem [6, 14], where the key issue is to look for low-cost plans with high success probabilities for customers based on previous experience. These plans can be generated on a case-by-case basis as in the previous situation for Basil and Steve, or the plans can be a single strategy that is applicable for a subset of customers; for example, a decision might be to send gives to all customers whose income is over $50,000.00 a year. The actions are only given implicitly in the form of attributes and their combinations, and the effects of actions can only be discovered statistically.

In this paper, we present a novel formulation of the above marketing plan problems for AI planning. We explore a case-based planning solution to the problem, where the case bases are extracted from a large raw customer database using data mining techniques. We consider the overall utility of the marketing plans developed, and propose solutions that provide tradeoff between quality of solution and speed of computation.

2. Understanding the Problem

The marketing strategy-planning problem departs from traditional planning significantly. First, the goals are not clearly given in a logical manner, as is done in many other planning algorithms. The goal in marketing is to increase the overall profit of a company while keeping the cost low. This goal has to be translated to individual actions for each customer. A second difference from the traditional planning is that actions are not given in the traditional way. Instead of starting out with a well-defined set of actions schemata, in business marketing the actions are only implicitly given. These actions must be constructed as the marketing strategy takes shape. For example, in direct marketing in a cell phone company, the effects of actions such as reducing the customer fees can only be measured when all customers' responses are known in the end. Finally, the marketing strategic plans themselves are not necessarily partially ordered action sequences. Instead, they are a set of actions on a segment of customers or on all customers that change the attribute values of a database.

We formulate the problem as a combination of data mining and case-based planning problems. In this approach, we first identify typical positive cases from a large dataset to form a case base, and then use the case base formulate the actions that adapt each incoming problem by finding its nearest neighbor in the case base.

More specifically, we first classify the training data into two classes: the "good" data set contains data that belong to customers who have already been accepted into the good class and the bad set those who have not. Given this labeled dataset, our second step is to perform a clustering analysis to find out a number of representative good cases of customers that can be "role models" for the rest and that represent the centroids of the good class distribution. We also identify a subset of highly relevant and actionable attributes of the database table that can be used to generate actions. The relevant actions are derived based on a feature extraction algorithm. The actionable attributes help project both the good and the bad databases on this set of attributes.

The representative data points discovered by data mining comprise a case base. For each new customer in the testing data, we compute a nearest neighbor from the case base for each customer in bad class. The new customers are then given advise on what actions are needed to transform themselves to a good case.

There are two important issues in this approach. The first issue is how to construct a concise case base from a large database. In this paper, we consider three approaches. The most naïve one is to simply use the original database as the case base. While this model allows the creation of optimal plans from the past data, this approach is highly inefficient. The second approach constructs clusters from the database, and takes the centroids of the clusters as the potential cases for the case base. This approach can be very efficient, but the quality of the cases is still not optimized. This is because in creating role models for the positive class, it is more desirable to find cases that are "close" to the majority of the negative instances. These cases are often located near the "boundary" of the distributions of these classes. In order to find these boundary cases, our third approach is to apply a support vector machine-learning algorithm for extracting the support vectors as cases. These cases can give rise to more cost-effective plans.

A second important issue is that for each advisee, how to select the target role model for advice generation. Here we consider two approaches. The first is to apply a nearest neighbor algorithm, which computes the distance between cases from the cost of actions attached to the attributes. While this approach provides plans that minimize the total costs, it does not give advice on the success likelihood of the plans. In practice, it is not always guaranteed that a switching plan will work. The probability of success is highly dependent on the distribution of classes around a role model. Thus, our second approach is to consider the utility of each potential role model, taking into account both the cost of switching and the probability of success. We show that this approach provides a much better advice plan for the customers.

From an AI planning perspective, this paper raises several challenging issues. First, instead of considering a well-defined goal, we need to consider the problem of finding goals to achieve. These goals correspond to role models in our case base. Second, instead of finding plans for a single user, we need to find plans for massive users and consider their overall gain. Third, instead of given logical representations of actions ahead of time, we have to compose these actions statistically. Finally, instead of delivering the plans for a robotic system to execute, our plans are in the form of advises for users to follow. These advises have immediate applications in the financial and marketing applications.

Our approach is related to several existing areas of research. The first is data mining and machine learning. In this area, researchers are interested in building statistical models of the database for classification and data analysis [4, 5, 8]. A typical statistical model partitions the test data into different classes according the trained model learned from the training data. A large literature exists in this area, such as work on decision tree analysis [8] association analysis [11] and Bayesian network models.

A second area of research is case-based reasoning, in particular case-base maintenance [9, 10] and case-based planning [14]. In the case-based planning work of Veloso [14], a new problem is solved by consulting a case base of past solutions. A new solution is then formed by deriving the difference between a past solution and the new problem by applying case-adaptation methods. However, to apply case-based planning to our problem, a case base have to be known and the plans in the case base have to be logically well formed. We don't have this luxury in marketing strategy planning, because the case base have to be discovered from the customer data first. Related to this issue is the so-called case-base maintenance problem, which has received much attention lately. In case-base maintenance [10] the focus has been on how to update case bases from problem solution pairs, and not on least-cost case-base generation for class-transformation.

In AI Planning, the objective has been to find a sequence or sequences of actions to achieve a user specified goal or objective. In our situation, the goal to be achieved is only implicitly given; that is, the goal is to enable a customer to become eligible for credit loan. The plans are to convert all the bad customers into good ones, using the case base as guide. This objective is related work in decision theory [12, 13], but the problem of scaling up using a small case base has not been addressed. The key issue again is to find a good case base from the database so that the overall cost of negative-case transformation is minimized.

3. Case-Base Mining By Clustering

We formulate the case-mining problem formally. Given a database of customer records, we assume that each customer record is labeled as either a positive or negative class. Multiple class generalization is possible but will be considered separately in future work. Each attribute is labeled either as actionable or non-actionable. For each actionable attribute A and values v1 and v2 of A, there is a cost function: *cost (A, v1, v2)* which is a real value.

The problem to be solved is to find a case base of *K* positive instances, such that the total cost of converting from all negative instances in the test set to their nearest neighbors in the case base is minimal. We will present two approaches to the problem, one requiring that the use specify the value for *K*, and the other does not.

In the extreme situation, the value of *K* is equal to the size of the population of all positive instances. In this case, the system is able to find the optimal solution, where each bad case is paired with its closest positive case. The drawback of this extreme situation is that the computational cost for planning for each individual negative instance is too large; in the realistic situation, the database may contain millions of customers. Finding the optimal solution for all negative instances is neither feasible nor necessary in practice.

Our first step is to find *K* near-optimal instances to populate the case base. When K is not known in advance, we can apply a second method discussed below to find the optimal cases. This case base will consist of *K* instances that are highly representative of the distribution of the customer information in the original database. In this work, we use all positive instances as training data for the case base model, and the negative instances as testing data set for evaluation. To see the interplay of efficiency with the size of case base, we show experimental results of quality of advice versus the time to come up with the switching plans of a certain quality. The case base quality is defined as the total cost to transform all negative instances into a positive counterpart in the testing set.

Our first case-base mining algorithm is described in more detail in Table 3. Given an input database, we divide the database into a training database and a testing database. The training database consists of the positive instances of the original database, whereas the testing data are the negative instances.

Table 3. **Algorithm** *Centroids-CBMine* (database DB, int K)

Steps	**Begin**
1	*casebase*= emptyset;
2	DB = *RemoveIrrelevantAttributes*(DB);
3	Separate the DB into DB+ and DB-;
4	Clusters+ = *ApplyKMeans*(*DB+*, *K*);
5	**for each** *cluster* in Clusters+, **do**
6	C = *findCentroid*(cluster);
7	*Insert*(C, *casebase*);
8	**end for**;
9	Return *casebase*;
	End

In the algorithm *Centroids-CBMine* in Table 3, the input database is DB. There are two classes in this database, where the positive class corresponds to population of desired cases and the negative class the unconverted cases. Step 2 of the algorithm performs feature extraction by applying a feature filter to the database to remove all attributes that are considered low in information content. For example, if two attributes *A1* and *A2* in the database are highly correlated, then one of them can be removed. Similarly, if an attribute *A* has very low classification power for the data, then it can be removed as well. In our implementation, we apply a C4.5 decision-learning algorithm to the database DB. After a decision tree is constructed, the attributes that are not contained in the tree are removed from the database; these are the irrelevant attributes.

Step 3 of the algorithm separates the training database into two partitions, a positive-class subset and a negative-class subset. Step 4 of the algorithm performs the K-means clustering on the positive-class sub-database [2]. K-means finds *K* locally optimal centroids by repeatedly applying the *EM* algorithm on a set of data. Other good clustering algorithms can also be used here in place of K-means. Step 6 of the algorithm finds centroids of the K clusters found in the previous step. These centroids are the bases of the case base constructed thus far, and are returned to the user. Finally, Step 9 returns the case base as the output.

Once the case base is built, it can then be applied to a set of testing negative-class cases to see what the total cost would be for converting all the negative cases to positive ones. For each negative class case C1 in the test data set, a one-nearest neighbor algorithm is applied to the case base to find the most similar case C2. The difference between C1 and C2 are used to generate the switch plan.

A critical issue for this approach is the tradeoff among the size of the case base, the quality of the model built and the time taken to build the case-base model. The quality measure is defined in terms of the total cost of converting negative instances to positive ones based on the basic cost elements of performing a conversion from one value to another for the actionable attributes. Recall that these elementary cost functions are *cost(A, v1, v2)*, which is a real value denoting the cost of switching attribute A from value v1 to value v2. Then, the cost of the model on an entire population of test data is the sum of all costs for all actions on each datum in the testing set. Assuming that the *j*th attribute for an *i*th customer is A_{ij} , Equation (1) shows the cost formula.

$$Cost = \sum_{i=1}^{|DB|} \sum_{j=1}^{l} \cos t(A_{ij}, v_1, v_2) \qquad (1)$$

The specification of the cost of switching an attribute A_{ij} from v1 to v2 can depend more than the attribute and its two values; it can in fact depend on the context of the switching. In this paper we simplify this consideration by only considering the attribute itself; but this restriction can be relaxed later.

4. Case-Base Mining by Support Vector Machines

The centroid-based case-mining method extracts cases from the positive-class cluster centroids and takes into account only the positive class distribution. By considering the distribution of both the positive and negative class clusters, we can do better.

The key issue then is to identify the positive cases on the *boundary* between the positive and negative cases, and select those cases as the final ones for the switching-plan generation. The cases along the boundary hyper-plane correspond to the support vectors found by an SVM classifier [3, 7]. These cases are the instances that are closest to the maximum margin hyper-plane in a hyperspace after an SVM system has discovered the classifier [3].

By exploiting the above idea, we have a different case-mining algorithm, SVM-*CBMine()*. In the first step, we perform SVM learning on the database to locate the support vectors. Then we find the support vectors and insert them into the case base. This algorithm is illustrated in Table 4.

Compared with the Centroids-CBMine algorithm, the SVM-CBMine algorithm has several advantages. First, because the cases are the support vectors themselves, there is no need to specify the input parameter *K* as in the Centroids-CBMine algorithm; the parameter *K* is used to determine the number of clusters to be generated in K-means. This corresponds to parameter-less case mining. Second, because the cases are themselves the boundary cases, they are naturally better examples for the entire negative-class members to switch to; the costs would be lower. However, the SVM based methods have also their drawbacks. A potential drawback is that SVM based classifiers are usually very costly to generate and are highly dependent on the data distribution. As we will see below in the experimental section, there are many datasets for which the SVM classification fails to produce any result within a limited amount of time, resulting in no case bases at all. As we will also point out, such situations occur when there is no clear boundary between the two classes.

Table 4. **Algorithm** *SVM-CBMine* (database *DB*, int *K*)

Steps	Begin
1	*casebase* = Emptyset;
2	*Vectors* = *SVM*(*DB*);
3	**for each** positive support vector C in *Vectors* **do**
4	Insert(C, *casebase*);
5	**end for**
6	Return casebase;
	End

5. Experimental Results

Our experiments are aimed at finding out the tradeoff among the system execution time, which is the model-building time plus the model-application time on test cases, the size of the model (the number of cases) and the total cost of switching plans for converting all negative examples into positive ones. We are also interested in the effect of distribution of the training data on the model quality. In our experiments we tested on both artificially generated data and some real data sets. The comparison made here did not use any attribute filtering algorithm to remove the irrelevant attributes. The experiments are performed on an Intel PC with one Gigahertz CPU.

We first tested the algorithms on a artificial data set generated on a two-dimensional space (x, y), using a Gaussian distribution with different means and co-variance matrix for the + and the – classes. When the means of the two distributions are separated, we expected the class boundaries are easy to identify by the

SVM-based method (see Figure 1). On the other hand, when the two means are very close to each other, there will not be an easy-to-find boundary; in this case the centroid-based method will perform better. The cost of switching a negative case to a positive one is defined as the Euclidean distance on the (x, y) plane.

In this distribution, the mean for the positive case distribution is *mean1*=(7, 8), with a co-variance matrix [(0.6, 0.3), (0.3, 1.8)]. For the negative class, the location of the mean *mean2* moves from being far away from the *mean1* to being close to it. The co-variance matrix for the positive class is defined as [(0.8, -0.5) (-0.5, 3.2)]. Table 5 shows the test results. In this table, SVM stands for the *SVM-CBMine* result. SV=3 indicates that three support vectors were found to populate the case base. Parameter *K* indicates the number of clusters generated by K-means algorithm for the *Centroids-CBMine()* system. In the table, the time, size and cost values are all indicated. Finally, Optimal (last row) indicates the cost and time for the model using all positive examples as the cases in the case base. Similarly, a second distribution is listed in Table 6, corresponding to the situation when the centers for the negative distribution are moved closer to that of the positive distributions. As can be seen from the progression of the data distribution, as the two classes are distributed farther apart from each other, the SVM-based method is a clear winner. This is because it uses far less time than the optimal method, and yet its total cost is nearly the same as that of the optimal method. As can be seen from the K-means based method, as the number *K* of clusters increases, the cost of switching plans also decreases. However, the time it takes to build and execute the model also increases with *K*. On the other hand, as the two distributions move close to each other such that there are no clear boundaries, as in the case of Table 6, the SVM method selects nearly all the positive examples as cases in the case base, rendering it useless. Thus, its time expense is also very high. In this case, the K-means based method is recommended.

Table 5. Result for *mean2* = (3, 4) (See Figure 3)

	Cost	Time (s)
K = 10	1934.2	1.3
K = 50	1464.6	6.6
K = 100	1420.8	13.8
SVM SV = 3	1225.6	0.9
Optimal	1220.0	7.03

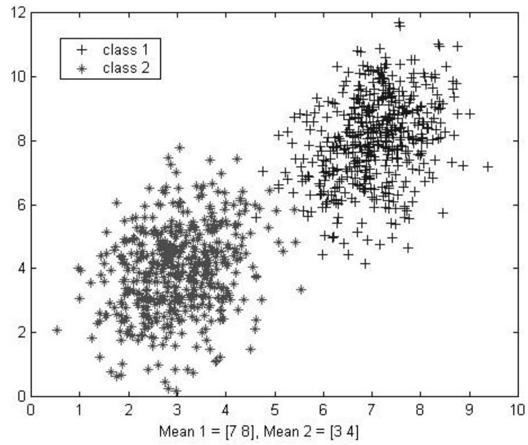

Figure 1. Distribution of the class 1=positive (+) and class 2=negative (*) data.

Table 6. Result for ***mean2*** = [7 8]

	Cost	Time (s)
K = 10	259.3	1.3
K = 50	133.4	6.5
K = 100	101.6	12.0
SVM SV = 487	58.5	25.6
Optimal	56.7	8.5

We next compared the three models on the some UCI data sets [1]. In Table 7, which is the data from German credit approval data, the SVM based method is unable to find any clear boundaries between the two classes; thus in this case the K-means based method wins. SVM cannot produce any case base either for other UCI data sets including the adult Database, the Liver-disorders Database, etc.

Table 7. German Credit dataset: 20 attributes, 2 classes 1000 instances (700 +, 300 -)

	Cost	Time (s)
K = 10	1211.2	3.3
K = 50	1108.9	7.6
K = 100	1060.8	14.9
SV = 700	934.6	18.8
Optimal	934.6	17.5

6. Utility Guided Plan Generation

The previous sections solved the plan generation problem using a nearest neighbor approach. The plan used to advice a customer is one that is associated with the least cost. While this is guaranteed to generate a cost effective plan, it is not guaranteed to generate a plan that will achieve its intended target all the time. In reality, the positive and negative cases are often distributed in a mixed manner. Several negative cases may surround a positive case. When executing a customer-switching plan, it is likely that the customer following the plan will land on a wrong target; it is wrong because it corresponds to a "unreliable" positive case whose neighborhood is dominated by negative instances, rendering the switching low probability of success. A more sensible method will consider not only the cost of switching, but also the probability of success of each switching.

We can estimate the probability of success of switching to a certain target to be the probability density of positive instances around a target. More formally, let $p(+|t)$ be the probability density of an instance t, $cost(x, t)$ be the cost of switch from x to target case t, and $maxCost$ be the maximum value among the different costs of switching from x to every possible case y in the case base. The utility function we use for ranking cases in a case base is defined in Equation (2) below. The target case t with the maximum rank is chosen as the role model for switching-plan generation for customer x.

$$rank(x,t) = p(+\,|\,t) - \frac{\cos t(x,t)}{\max Cost} \qquad (2)$$

Finally, we performed a scale-up test using the by IBM QUEST synthetic data generator. We generated the training dataset with nine attributes, 50% positive class and 50% negative class. An excerpt of the database is shown in Table 8. Our results are shown in Table 9. It is clear from the table that with large data, the centroid-based method is able to scale up much better than the SVM based method.

Table 8. An excerpt from the synthetic dataset.

Salary	Commission	Education	Car	…
65498	49400	1	2	…
24523	0	2	3	…
78848	0	2	6	…
74340	29463	0	3	…
42724	0	1	4	…

Table 9. CPU-time comparison of Centroid-based Method and SVM-based method.

Log$_{10}$(N), ***N*=Database size**	**CPU Time (Seconds)**	
	Centroid-based	**SVM**
2	0.660	0.650
2.5	1.640	1.870
3	6.480	14.330
3.5	23.120	319.180
4	95.850	3,834.900
4.5	312.970	No result in 5 hrs
5	1,938.430	No result in 7 hrs

7. Conclusions and Future Work

In this paper, we proposed a case-based solution for the switching-plan generation, a problem that arises in business planning. The central issue of the problem lies in the discovery of high-quality case bases from a large data set. This problem corresponds to finding goals to achieve for each customer. We proposed two solutions for the problem. For the data distribution where the two classes are clearly separated, the SVM-CBMine algorithm, which is an SVM-based method, should be used. When the data distributions are not separated well by a boundary, the cluster-centroids based method is recommended. Furthermore, we compared the solutions where plan generation is done based on distance alone and the solution where the probability of success is also taken into account. It was shown that the solution with utility consideration is superior. In addition, the centroid-based method is shown to scale much better than the SVM-based method, demonstrating a quality-speed tradeoff.

In the future, we will continue to explore other forms of case mining and the related problem of switch-plan generation. Other cost functions will be considered. Attributes weights will be taken into account as well.

Acknowledgment

We thank Hong Cheng and Ke Wang discussions and assistance with the experiments. The research is supported by a grant from Hong Kong University of Science and Technology.

8. References

[1] C. L. Blake, C.J. Merz (1998). *UCI Repository of machine learning databases* Irvine, CA: University of California, Department of Information and Computer Science. http://www.ics.uci.edu/~mlearn/MLRepository.html

[2] P. S. Bradley and U. M. Fayyad. *Refining initial points for **k-means** clustering*. In Proceedings of the Fifteenth International Conference on Machine Learning (ICML '98), pages 91--99, San Francisco, CA, 1998. Morgan Kaufmann.

[3] G. C. Cowley. *MATLAB Support Vector Machine Toolbox. v0.54B* University of East Anglia, School of Information Systems, Norwich, Norfolk, U.K. NR4 7TJ, 2000. http://theoval.sys.uea.ac.uk/~gcc/svm/toolbox

[4] C. X. Ling and C. Li. *Data mining for direct marketing: Problems and solutions*. In Proceedings 4th International Conference on Knowledge Discovery in Databases (KDD-98), New York, 1998.

[5] P. Domingos and M. Richardson. Mining the Network Value of Customers. Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. August 2001. ACM. N.Y. N.Y. USA

[6] D. Leake. *Case-based Reasoning -- Experiences*, Lessons and Future Directions. AAAI Press/ The MIT Press, 1996.

[7] J. C. Platt, *Fast training of support vector machines using sequential minimal optimization*, in Advances in Kernel Methods - Support Vector Learning, (Eds) B. Scholkopf, C. Burges, and A. J. Smola, MIT Press, Cambridge, Massachusetts, chapter 12, pp 185-208, 1999.

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

[9] B. Smyth and M. T. Keane. *Remembering to forget: A competence--preserving deletion policy for case--based reasoning systems*. In Proceedings of the 14th International Joint Conference on Artificial Intelligence, pp 377--382, 1995

[10] *Computational Intelligence Journal, Special Issue on Case-base Maintenance*. Blackwell Publishers, Boston MA UK. Vol. 17, No. 2, May 2001. Editors: D. Leake, B. Smyth, D. Wilson and Q. Yang.

[11] R. Agrawal and R. Srikant. *Fast algorithm for mining association rules*. Proceedings of the Twentieth International Conference on Very Large Databases. 1994. pp 487-499

[12] F. Bacchus and A. Grove, *Graphical Models for Preference and Utility*, Proceedings of the Uncertainties in AI 1995.

[13] R. L. Keeney and H. Raiffa, *Decisions with Multiple Objectives: Preferences and Value Trade-offs*, Wiley, New York, 1976

[14] M. Veloso *Planning and learning by analogical reasoning*. Number 886 in Lecture Notes in Artificial Intelligence. Springer Verlag. 1994

[15] Q. Yang. *Intelligent Planning*. Springer Verlag. 1997