Classification of Imbalanced Marketing Data with Balanced Random Sets

Vladimir Nikulin V.NIKULIN@UQ.EDU.AU
Department of Mathematics, University of Queensland, Australia

Geoffrey J. McLachlan GJM@MATHS.UQ.EDU.AU
Department of Mathematics, University of Queensland, Australia

Editor: Gideon Dror, Marc Boullé, Isabelle Guyon, Vincent Lemaire, David Vogel

Abstract

With imbalanced data a classifier built using all of the data has the tendency to ignore the minority class. To overcome this problem, we propose to use an ensemble classifier constructed on the basis of a large number of relatively small and balanced subsets, where representatives from both patterns are to be selected randomly. As an outcome, the system produces the matrix of linear regression coefficients whose rows represent the random subsets and the columns represent the features. Based on this matrix, we make an assessment of how stable the influence of a particular feature is. It is proposed to keep in the model only features with stable influence. The final model represents an average of the base-learners, which is not necessarily a linear regression. Proper data pre-processing is very important for the effectiveness of the whole system, and it is proposed to reduce the original data to the most simple binary sparse format, which is particularly convenient for the construction of decision trees. As a result, any particular feature will be represented by several binary variables or bins, which are absolutely equivalent in terms of data structure. This property is very important and may be used for feature selection. The proposed method exploits not only contributions of particular variables to the base-learners, but also the diversity of such contributions. Test results against KDD-2009 competition datasets are presented.

Keywords: ensemble classifier, gradient-based optimisation, boosting, random forests, decision trees, matrix factorisation

1. Introduction

Ensemble (including voting and averaged) classifiers are learning algorithms that construct a set of many individual classifiers (called base-learners) and combine them to classify test data points by sample average. It is now well-known that ensembles are often much more accurate than the base-learners that make them up (Biau et al., 2007). The tree ensemble called "random forests" (RF) was introduced in (Breiman, 2001) and represents an example of a successful classifier. In another example, the bagged version of the support vector machine (SVM) (Zhang et al., 2007) is very important because direct application of the SVM to the whole data set may not be possible. In the case of the SVM, the dimension of the kernel matrix is equal to the sample size, which thus needs to be limited.

Our approach was motivated by (Breiman, 1996), and represents a compromise between two major considerations. On the one hand, we would like to deal with balanced data. On the other hand, we wish to exploit all available information. We consider a large number n

of balanced subsets of available data where any single subset includes two parts (a) nearly all 'positive' instances (minority) and (b) randomly selected 'negative' instances. The method of balanced random sets (RS) is general and may be used in conjunction with different base-learners.

Regularised linear regression (RLR) represents the most simple example of a decision function. Combined with quadratic loss function it has an essential advantage: using a gradient-based search procedure we can optimise the value of the step size. Consequently, we will observe a rapid decline in the target function.

By definition, regression coefficients may be regarded as natural measurements of influence of the corresponding features. In our case we have n vectors of regression coefficients, and we can use them to investigate the stability of the particular coefficients.

Proper feature selection (FS) may reduce overfitting significantly. We remove features with unstable coefficients, and recompute the classifiers. Note that stability of the coefficients may be measured using different methods. For example, we can apply the t-statistic given by the ratio of the mean to the standard deviation (Nikulin, 2008).

Matrix factorisation, an unsupervised learning method, is widely used to study the structure of the data when no specific response variable is specified. In principle, it would be better to describe the data in terms of a small number of meta-features, derived as a result of matrix factorisation, which could reduce noise while still capturing the essential features of the data. In addition, latent factor models are generally effective at estimating overall structure that relates simultaneously to most or all items.

Note that the methods for non-negative matrix factorisation (NMF) which were introduced in (Lee and Seung, 2001) are valid only under the condition that all the elements of all input and output matrices are non-negative. In Section 3.4 we formulate a general method for matrix factorisation, which is significantly faster compared with NMF. Note also that some interesting and relevant ideas for the stochastic gradient descent algorithm were motivated by methods used in the well-known Netflix Cup; see, for example, (Paterek, 2007).

The proposed approach is flexible. We do not expect that a single specification will work optimally on all conceivable applications and, therefore, an opportunity of tuning and tailoring the algorithm is important.

Results which were obtained during the KDD-2009 Data Mining Competition are presented.

2. Task Description

The KDD Cup 2009[1] offered the opportunity to work on large marketing databases from the French Telecom company Orange to predict the propensity of customers to switch provider (churn), buy new products or services (appetency), or buy upgrades or add-ons proposed to them to make the sale more profitable (up-selling).

Churn (Wikipedia definition) is a measure of the number of individuals or items moving into or out of a collection over a specific period of time. The term is used in many contexts, but is, most widely, applied in business with respect to a contractual customer base. For instance, it is an important factor for any business with a subscriber-based service model,

1. http://www.kddcup-orange.com/

including mobile telephone networks and pay TV operators. The term is also used to refer to participant turnover in peer-to-peer networks. Appetency is the propensity to buy a service or a product. Up-selling (Wikipedia definition) is a sales technique whereby a salesman attempts to have the customer purchase more expensive items, upgrades, or other add-ons in an attempt to make a more profitable sale. Up-selling usually involves marketing more profitable services or products, but up-selling can also be simply exposing the customer to other options that he or she may not have considered previously. Up-selling can imply selling something additional, or selling something that is more profitable or, otherwise, preferable for the seller instead of the original sale.

Customer Relationship Management (CRM) is a key element of modern marketing strategies. The most practical way in a CRM system to build knowledge on customer is to produce scores. The score (the output of a model) is computed using input variables which describe instances. Scores are then used by the information system (IS), for example, to personalize the customer relationship. There is also an industrial customer analysis platform able to build prediction models with a very large number of input variables (known as explanatory variables or features).

Generally, all features may be divided into two main parts: primary (collected directly from the customer) and secondary, which may be computed as a functions of primary features or may be extracted from other databases according to the primary features. Usually, the number of primary features is rather small (from 10 to 100). On the other hand, the number of secondary features may be huge (up to a few thousands). In most cases, proper design of the secondary features requires deep understanding of the most important primary features.

The rapid and robust detection of the features that have most contributed to the output prediction can be a key factor in a marketing applications. Time efficiency is often a crucial point, because marketing patterns have a dynamic nature and in a few days time it will be necessary to recompute parameters of the model using fresh data. Therefore, part of the competition was to test the ability of the participants to deliver solutions quickly.

3. Main Models

Let $\mathbf{X} = (\mathbf{x}_t, y_t), t = 1, \ldots, n$, be a training sample of observations where $\mathbf{x}_t \in \mathbb{R}^\ell$ is ℓ-dimensional vector of features, and y_t is binary label: $y_t \in \{-1, 1\}$. Boldface letters denote vector-columns, whose components are labelled using a normal typeface.

In supervised classification algorithms, a classifier is trained with all the labelled training data and used to predict the class labels of unseen test data. In other words, the label y_t may be hidden, and the task is to estimate it using vector of features. Let us consider the most simple linear decision function

$$u_t = u(\mathbf{x}_t) = \sum_{j=1}^{\ell} w_j \cdot x_{tj} + b,$$

where w_i are weight coefficients and b is a bias term.

We used AUC as an evaluation criterion, where AUC is the area under the receiver operating curve. By definition, ROC is a graphical plot of true positive rates against false positive rates.

3.1 RLR Model

Let us consider the most basic quadratic minimization model with the following target function:

$$L(\mathbf{w}) = \Omega(\phi, n, \mathbf{w}) + \sum_{t=1}^{n} \xi_t \cdot (y_t - u_t)^2 , \qquad (1)$$

where $\Omega(\phi, n, \mathbf{w}) = \phi \cdot n \cdot \|\mathbf{w}\|^2$ is a regularization term with ridge parameter ϕ; the ξ_t are non-negative weight coefficients.

Remark 1 *The aim of the regularization term with parameter ϕ is to reduce the difference between training and test results. Value of ϕ may be optimized using cross-validation; see Mol et al. (2009) for more details.*

3.1.1 Gradient-based optimisation

The direction of the steepest decent is defined by the gradient vector

$$g(\mathbf{w}) = \{g_j(\mathbf{w}), j = 1, \dots, \ell\},$$

where

$$g_j(\mathbf{w}) = \frac{\partial L(\mathbf{w})}{\partial w_j} = 2\phi \cdot n \cdot w_j - 2 \sum_{t=1}^{n} x_{tj} \xi_t \left(y_t - u_t\right).$$

Initial values of the linear coefficients w_i and the bias parameter b may be arbitrary. Then, we recompute the coefficients by

$$\mathbf{w}^{(k+1)} = \mathbf{w}^{(k)} + \delta_k \cdot g(\mathbf{w}^{(k)}),\ b^{(k+1)} = b^{(k)} + \frac{1}{n} \sum_{t=1}^{n} \xi_t \cdot (y_t - u_t),$$

where k is the iteration number. Minimizing (1), we find the size of the step according to the formula

$$\delta = \frac{L_1 - L_2 - \phi \cdot n \sum_{j=1}^{\ell} w_j g_j}{\sum_{t=1}^{n} \xi_t s_t^2 + \phi \cdot n \sum_{j=1}^{\ell} g_j^2},$$

where

$$L_1 = \sum_{t=1}^{n} \xi_t s_t y_t, \quad L_2 = \sum_{t=1}^{n} \xi_t s_t u_t, \quad s_t = \sum_{j=1}^{\ell} x_{tj} g_j.$$

3.2 Random Sets

According to the proposed method, we consider large number of classifiers, where any particular classifier is based on a relatively balanced subset with randomly selected (without replacement) ‘positive’ and ‘negative’ instances. The final decision function was calculated as the sample average of the single decision functions or base-learners.

Definition 2 *We refer to the above subsets as random sets $RS(\alpha, \beta, m)$, where α is the number of positive cases, β is the number of negative cases, and m is the total number of random sets.*

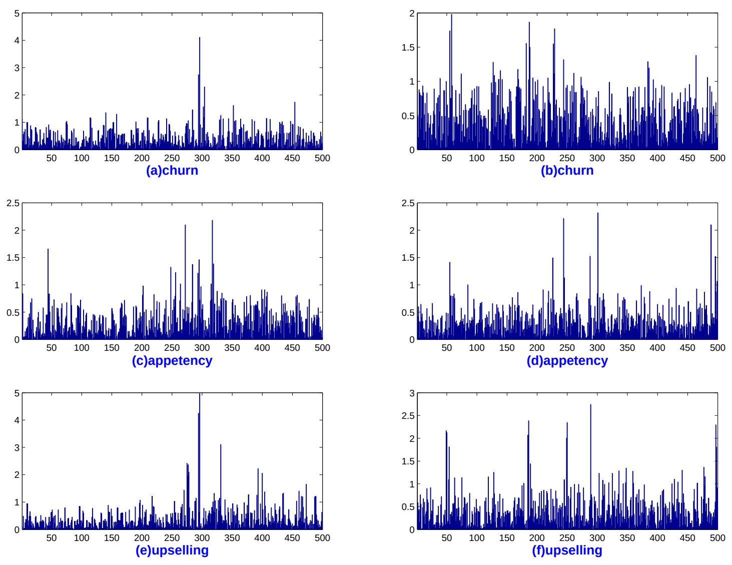

Figure 1: MVF, ratios (2): (a-b) Churn, (c-d) Appetency and (e-f) Upselling. In order to improve visibility, we displayed two fragments (out of 9586) with 500 ratios each.

This model includes two very important regulation parameters: m and $q = \frac{\alpha}{\beta} \leq 1$, where q is the proportion of positive to negative cases. In practice, m must be big enough, and q can not be too small.

We consider m subsets of $\mathbf{X}$ with α positive and $\beta = k \cdot \alpha$ negative data-instances, where $k \geq 1, q = \frac{1}{k}$. Using gradient-based optimization (see Section 3.1.1), we can compute the matrix of linear regression coefficients: $W = \{w_{ij}, i = 1, \ldots, m, j = 0, \ldots, \ell\}$.

3.3 Mean-Variance Filtering

The mean-variance filtering (MVF) technique was introduced in (Nikulin, 2006), and can be used to reduce overfitting. Using the following ratios, we can measure stability of the contributions of the particular features by

$$r_j = \frac{|\mu_j|}{\lambda_j}, \ j = 1, \ldots, \ell, \tag{2}$$

where μ_j and λ_j are the mean and standard deviation corresponding to the j-column of the matrix W.

A low value of r_j indicates that the influence of the jth binary secondary feature is not stable. We conducted feature selection according to the condition: $r_j \geq \gamma > 0$. The sum of the r_j corresponding to the original feature f will give us a stability rating for the feature f.

3.4 Learning from the Test Data with Matrix Factorisation

Unlike classification and regression, matrix decomposition requires no response variable and thus falls into the category of unsupervised learning methods. Using this fact as a motivation, we can merge training and test datasets into one matrix $\mathbf{X}^+$. There are possible differences between training and test sets, and we can expect that the matrix factorisation will be able to smooth such differences.

In this section, we describe the procedure for undertaking the matrix factorisation,

$$\mathbf{X}^+ \sim \mathbf{AB}, \tag{3}$$

where matrices $\mathbf{A} = \{a_{if}, i = 1, \dots, 2n, f = 1, \dots, k,\}$ and $\mathbf{B} = \{b_{fj}, f = 1, \dots, k, j = 1, \dots, \ell\}$. After the factorisation, the first n rows of the matrix $\mathbf{A}$ will used for training and the last n rows will be used for testing.

The matrix factorisation represents a gradient-based optimisation process with the objective to minimise the following squared loss function,

$$L(\mathbf{A}, \mathbf{B}) = \frac{1}{2n \cdot \ell} \sum_{i=1}^{2n} \sum_{j=1}^{\ell} E_{ij}^2, \tag{4}$$

where $E_{ij} = x_{ij} - \sum_{f=1}^{k} a_{if} b_{fj}$.

Algorithm 1 Matrix factorisation.

```
1: Input: $\mathbf{X}^+$ - dataset.
2: Select $\nu$ - number of global iterations; $k$ - number of factors; $\lambda > 0$ - learning rate,
   $0 < \tau < 1$ - correction rate, $L_S$ - initial value of the target function.
3: Initial matrices $\mathbf{A}$ and $\mathbf{B}$ may be generated randomly.
4: Global cycle: repeat $\nu$ times the following steps 5 - 17:
5: samples-cycle: for $i = 1$ to $2n$ repeat steps 6 - 17:
6: features-cycle: for $j = 1$ to $\ell$ repeat steps 7 - 17:
7: compute prediction $S = \sum_{f=1}^{k} a_{if} b_{fj}$;
8: compute error of prediction: $E = x_{ij} - S$;
9: internal factors-cycle: for $f = 1$ to $k$ repeat steps 10 - 17:
10: compute $\alpha = a_{if} b_{fj}$;
11: update $a_{if} \Leftarrow a_{if} + \lambda \cdot E \cdot b_{fj}$ (see (5a));
12: $E \Leftarrow E + \alpha - a_{if} b_{fj}$;
13: compute $\alpha = a_{if} b_{fj}$;
14: update $b_{fj} \Leftarrow b_{fj} + \lambda \cdot E \cdot a_{if}$ (see (5b));
15: $E \Leftarrow E + \alpha - a_{if} b_{fj}$;
16: compute $L = L(\mathbf{A}, \mathbf{B})$;
17: $L_S = L$ if $L < L_S$, and $\lambda \Leftarrow \lambda \cdot \tau$, otherwise.
18: Output: $\mathbf{A}$ and $\mathbf{B}$ - matrices of latent factors.
```

The above target function (4) includes in total $k(2n + \ell)$ regulation parameters and may be unstable if we minimise it without taking into account the mutual dependence between elements of the matrices $\mathbf{A}$ and $\mathbf{B}$.

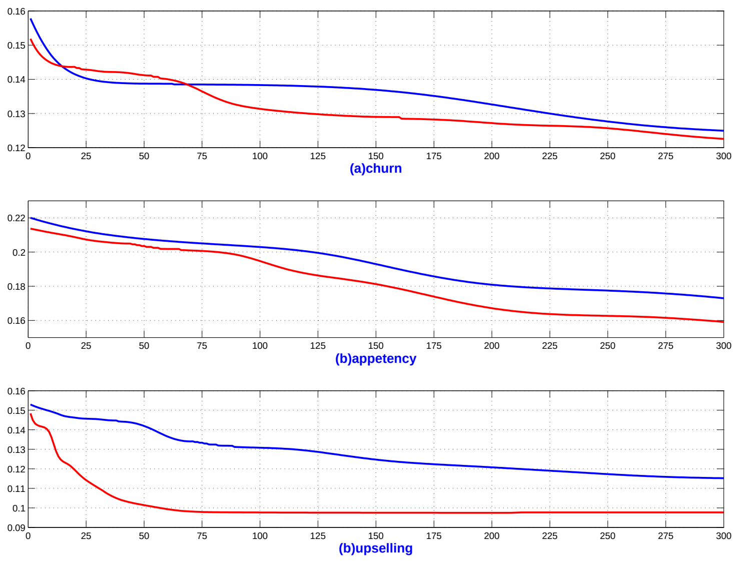

Figure 2: Convergence of Algorithm 1: (a) Churn, $\ell = 477$; (b) Appetency, $\ell = 532$; and (c) Upselling, $\ell = 385$, where blue lines correspond to $k = 8$ and red lines correspond to $k = 20$.

As a solution to the problem, we can cycle through all the differences E_{ij}, minimising them as a function of the particular parameters which are involved in the definition of E_{ij}. Compared to usual gradient-based optimisation, in our optimisation model we are dealing with two sets of parameters, and we therefore mix uniformly updates of these parameters, because the latter are dependent.

The following partial derivatives are necessary for Algorithm 1:

$$\frac{\partial E_{ij}^2}{\partial a_{if}} = -2E_{ij}b_{fj}, \tag{5a}$$

$$\frac{\partial E_{ij}^2}{\partial b_{fj}} = -2E_{ij}a_{if}. \tag{5b}$$

Similarly, as in Section 3.1, we can optimise here the value of the step-size. However, taking into account the complexity of the model, it will be better to maintain fixed and small values of the step size or learning rate. In all our experiments, we conducted matrix factorisation with the above Algorithm 1 using 300 global iterations with the following regulation parameters: $\lambda = 0.01$ - initial learning rate, $\xi = 0.75$ -correction rate. We conducted experiments with Algorithm 1 against the datasets in a binary format, and Figure 2 illustrates convergence of the algorithm.

4. Experiments

4.1 Pre-processing Technique

The sizes of the training and test datasets are the same and are equal to 50000. There are 14740 numerical and 260 categorical features in the large dataset. The training data are very imbalanced. The numbers of positive cases were 3672 (Churn), 890 (Appetency) and 3682 (Upselling) out of a total number of 50000.

Firstly, we conducted a basic checking of the categorical data. We detected 72 variables with only one value. In addition, we removed 74 variables, where the number of missing variables was greater than 49500. The number of the remaining variables was 184. As a next step, we considered all possible values for the latter 184 variables, and found that 1342 values occurred frequently enough to be considered as independent binary variables (otherwise, values were removed from any further consideration).

An effective way to link information contained in numerical and categorical variables is to transfer the numerical variables to binary format (as it was before in the application to the categorical variables). We used a technique similar to that used before in converting the categorical variables to binary format. We removed all variables with numbers of missing and zeros greater than 49500. The number of the remaining variables was 3245. Next, we split the range of values for any particular variable into 1000 subintervals, and computed the numbers of occurrences for any subinterval. These numbers were considered later as weights of the bins.

Then, we split all subintervals for the particular variable into 10 consecutive bins with approximately the same size (in terms of weights). In many cases, where weights of some subintervals were too big, the numbers of bins were smaller than 10.

Finally, we got a totally binary dataset in a sparse format with 13594 variables. After secondary trimming, we were left with 9586 binary features.

Remark 3 *It is a well-known fact that the exact correspondence between small and large datasets may be found. We managed to find such a correspondence as some other teams (it was rather a response to the findings of other teams). However, we can not report any significant progress (in the sense of the absolute scores), which was done by the help of this additional information.*

4.2 Results

The initial experiments, which were conducted against the vector of toy labels, were interesting and helpful for further studies. The system clearly detected all binary variables, which are secondary to the only one important original variable *N5963* (see Table 1). The definition of the transformation function between two known variables is a rather technical issue, which may be solved easily using two steps procedure: (1) sorting according to the explanatory variable, and (2) smoothing using sample averages in order to reduce noise.

As a first step (after labels for the Churn, Appetency and Upselling were released), we applied regularised linear regression model as described in Section 3.1. The number of the random sets was 100 and the ridge parameter was $\phi = 0.01$. In order to form any random set, we used about 90% of positive cases and $k = 1$. That is, any random set contains

Table 1: Training and test in terms of AUC with averaged score 0.8059 (initial results); 0.8373 (fast and slow tracks results); 0.8407 (best results); 0.851 (post-challenge results). The column FS indicates the number of variables, which were used in the model, where by $\star$ we indicate the number of binary variables

Status	Data	Method	Train	Test	FS
Initial	Churn	RLR	0.8554	0.7015	$9586^\star$
Initial	Appetency	LinearSVM	0.9253	0.8344	$9586^\star$
Initial	Upselling	RLR	0.9519	0.8819	$9586^\star$
Initial	Toy	RLR	0.7630	0.7190	$645^\star$
Fast	Churn	LogitBoost	0.7504	0.7415	39
Fast/Best	Appetency	BinaryRF	0.9092	0.8692	$145^\star$
Fast	Upselling	LogitBoost	0.9226	0.9012	28
Slow/Best	Churn	LogitBoost	0.7666	0.7484	41
Slow	Appetency	LogitBoost	0.9345	0.8597	33
Slow	Upselling	LogitBoost	0.9226	0.904	54
Best	Upselling	LogitBoost	0.9208	0.9044	44
Best	Toy	Special	0.7354	**0.7253**	1 (N5963)
Post	Churn	Ensemble	0.8772	**0.7618**	278
Post	Appetency	Ensemble	0.9586	**0.8835**	348
Post	Upselling	Ensemble	0.9628	**0.9077**	285

equal number of positive and negative instances. Note that in the case of Appetency, we considered initially the use of the SVM with a linear kernel; see the first 3 lines of Table 1.

Further, we employed mean-variance filtering, and reduced the number of features to 145 for Appetency, 151 for Churn, and 68 for Upselling (see Figure 1).

The participants received feedback against 10% of the test dataset (named leaderboard). In addition, we used cross-validation (CV) with up to 20 folds. Any CV-fold was formed under the strict condition that relation of the patterns is exactly the same as in the training dataset. Based on our CV-experiments, we observed a close relationship between the leaderboard and CV-results.

Remark 4 *After publication of the final results, we found that the relationship between the leaderboard and test results is also tight. It appears that in this particular case an "excessive" submissions against the leaderboard dataset did not lead to overfitting of the model. This prior knowledge may be very helpful for the second (slow) part of the competition.*

The binary (sparse) format gives a significant advantage in the sense of computational speed. But, it is not very important for the R-based packages in difference to the memory allocation. Accordingly, we returned to the original variables by replacing the binary features by their sequential indices (within the group corresponding to the particular original feature) before loading the new datasets into the R-environment.

We used mainly in our experiments five models RLR, LinearSVM, BinaryRF, LogitBoost and RF, where the last two models were implemented in R, the other models were written in C. For example, the following settings were used for the BinaryRF (Appetency case, see Table 1): (1) decision trees with up to 14 levels; (2) the number of features were selected randomly out of the range between 12 and 17 for any particular split; (3) the splitting

Table 3: Results of the winners

Track	Team	Churn	Appetency	Upselling	Score
Fast	IBM Research	0.7611	0.883	0.9038	0.8493
Slow	Uni. Melb.	0.7542	0.8836	0.9047	0.8475
Overall	-	0.7651	0.8836	0.9092	0.8526

process was stopped if improvement was less than 0.1% or number of data in the node was less than 100; 4) number of RS was 100 and number of trees for any RS was 400 (that means, the total number of trees was 40000).

5. Post-challenge Submissions

Firstly, we decided to rebuild completely all the databases. It was an extension of the previous databases based on the MVF feature selections ratings. Also, we took into account ratings from the KDD-preprint of the University of Melbourne team - winner of the slow track (see Table 3). We were working in two mutually dependent directions: binary databases for our own software binaryRF, and the corresponding integer databases of indexes for R packages named randomForest, ADA and GBM (for the last one we would like to thank the University of Melbourne team again, as they reported this package in their KDD-preprint).

Table 2: Selected pure results

Data	Model	Test	Weight	Data	Model	Test	Weight
Appetency	BinaryRF	0.8784	10	Churn	GBM	0.7613	-
Appetency	GBM	0.878	9	Upselling	GBM	0.9072	20
Appetency	ADA	0.8742	3	Upselling	ADA	0.9071	19

5.1 Ensemble Constructor

Using results of Table 2 separately we can achieve the score of 0.8489. Now, we shall describe a general framework how using ensemble technique we can increase the score to 0.851 based on the results of the above Table 2 and without any additional information. Note that the particular solutions, which were produced using different software may have approximately the same AUC, but they are very different in the structural sense and we can not link them directly.

Suppose we have k solutions, which may be represented by the squared matrix $\mathbf{T}$ with k columns. By $\mathbf{S}$ we shall denote the matrix, which was derived from the original matrix of solutions $\mathbf{T}$ by sorting on any column in an increasing order. In order to define the ensemble constructor we also need a matrix $\mathbf{R}$ of indices defining an exact correspondence between the elements of $\mathbf{T}$ and $\mathbf{S}$. More precisely, $s_{r_{ij}j} = t_{ij}$.

Let us denote by β_j the quality measurement of the solution with index $j = 1, \dots, k$. In order to simplify the following notation and without loss of generality we shall assume

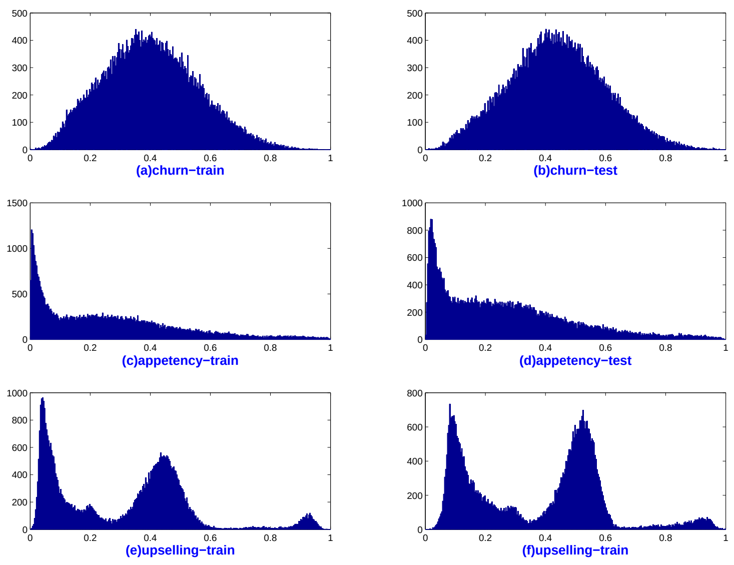

Figure 3: Histograms with 300 bins illustrating the structural similarities between training and test trajectories, where the right columns corresponds to the score 0.8509: (a-b) Churn, (c-d) Appetency and (e-f) Upselling.

that (a) solution i is better comparing with solution j if $\beta_i > \beta_j$, (b) the top solution corresponds to the index 1, that means $\beta_1 \geq \beta_j, j = 2, \ldots, k$. Then, we can accept solution N1 as a base solution, and shall adjust remaining $k-1$ solutions according to the rule: $q_{ij} = s_{r_{ij}1}, i = 1, \ldots, n, j = 2, \ldots, k$. Note that the first column in the matrix $\mathbf{Q}$ coincides with the first column of the original matrix $\mathbf{T}$ (base solution).

We shall define vector-column of the weight coefficients $w_j = \psi(\beta_j), \sum_{j=1}^{k} w_j = 1$, where ψ is an increasing function. In line with the proposed method, the ensemble solution will be computed according to the formula: $f = \mathbf{QW}$.

We can report a significant progress with above technique in application to the Appetency case. Our ensemble solution was constructed using three pure solutions as it is displayed in Table 2, where the weight coefficients are shown without normalisation. Also, we achieved some modest progress in application to the Upselling case. However, in the case of Churn we were not able to improve GBM-solution using above technique.

On the other hand, we were trying to exploit mutual dependences between different cases. For example, higher combined score in relation to the Appetency and Upselling cases most likely indicates lower score in application to the Churn case:

$$new.score_C = old.score_C - c_1 score_A - c_2 score_U, \tag{6}$$

where c_1 and c_2 are non-negative constants. In fact, using above method (6) we managed to achieve some small improvement only in application to the Churn case. Possibly, some interesting results may be achieved using multinomial logistic regression (MLR) (Bohning, 1992), as this model explore a hidden dependencies between labels. In our case we have four different labels: 1) Churn, 2) Appetency, 3) Upselling and 4) Other. In the case of MLR, we shall be dealing with the 3ℓ-dimensional Hessian matrix of second derivatives. We can expect that the matrix factorisation, as it is described in Section 3.4, will be effective to reduce original dimensionality ℓ to k - number of the latent factors.

It is interesting to note that the histograms displayed on Figure 3 illustrate remarkable similarity between left (training) and right (test) columns. The structure of the histograms corresponding to the Upselling case is far from simple and it will be very important to find explanations in the terms of the particular features.

6. Concluding Remarks

The main philosophy of our method may be described as follows. We can not apply fairly complex modelling systems to the original huge and noisy database, which contains more than 90% of useless information. So we conducted the FS step with three very simple and reliable methods, namely RS, RLR, and MVF. As an outcome, we produced significantly smaller datasets, which are able to be used as an input for more advanced studies.

In general terms, we have found that our results are satisfactory, particularly, for the most important fast track. Based on the results of our post-challenge submission, we can conclude that significant progress may be achieved using more advanced pre-processing and feature selection techniques. Also, it will be a good idea not to rely completely on any particular model or software, and conduct cross-validation experiments with several different models. In this way, the performance of the final solution might be improved using various ensemble techniques.

References

G. Biau, L. Devroye, and G. Lugosi. Consistency of random forests and other averaging classifiers. *Journal of Machine Learning Research*, 9:2015–2033, 2007.

D. Bohning. Multinomial logistic regression algorithm. *Ann. Inst. Statist. Math.*, 44(1):197–200, 1992.

L. Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.

L. Breiman. Bagging predictors. *Machine Learning*, 24:123–140, 1996.

D. Lee and H. Seung. Algorithms for non-negative matrix factorisation. NIPS, 2001.

C. Mol, S. Mosci, M. Traskine, and A. Verri. A regularised method for selecting nested groups of relevant genes from microarray data. *Journal of Computational Biology*, 16(5):677–690, 2009.

V. Nikulin. Learning with mean-variance filtering, SVM and gradient-based optimization. In *International Joint Conference on Neural Networks, Vancouver, BC, Canada, July 16-21*, pages 4195–4202. IEEE, 2006.

V. Nikulin. Classification of imbalanced data with random sets and mean-variance filtering. *International Journal of Data Warehousing and Mining*, 4(2):63–78, 2008.

A. Paterek. Improving regularised singular value decomposition for collaborative filtering. pages 39–42. KDD, San Jose, 2007.

B. Zhang, T. Pham, and Y. Zhang. Bagging support vector machine for classification of SELDI-ToF mass spectra of ovarian cancer serum samples. In *LNAI*, volume 4830, pages 820–826. Springer, 2007.