Knowl Inf Syst (2015) 43:29–51
DOI 10.1007/s10115-013-0722-y

REGULAR PAPER

Feature-selection-based dynamic transfer ensemble model for customer churn prediction

Jin Xiao · Yi Xiao · Anqiang Huang · Dunhu Liu · Shouyang Wang

Received: 20 November 2012 / Revised: 11 November 2013 / Accepted: 04 December 2013 /
Published online: 16 January 2014

Abstract Customer churn prediction is one of the key steps to maximize the value of customers for an enterprise. It is difficult to get satisfactory prediction effect by traditional models constructed on the assumption that the training and test data are subject to the same distribution, because the customers usually come from different districts and may be subject to different distributions in reality. This study proposes a feature-selection-based dynamic transfer ensemble (FSDTE) model that aims to introduce transfer learning theory for utilizing the customer data in both the target and related source domains. The model mainly conducts a two-layer feature selection. In the first layer, an initial feature subset is selected by GMDH-type neural network only in the target domain. In the second layer, several appropriate patterns from the source domain to target training set are selected, and some features with higher mutual information between them and the class variable are combined with the initial subset to construct a new feature subset. The selection in the second layer is repeated several times to generate a series of new feature subsets, and then, we train a base classifier in each one. Finally, a best base classifier is selected dynamically for each test pattern. The

J. Xiao
Business School, Sichuan University, Chengdu 610064, China
e-mail: xiaojin@scu.edu.cn

Y. Xiao
School of Information Management, Central China Normal University, Wuhan 430079, China
e-mail: xybill@amss.ac.cn

A. Huang
School of Economics and Management, Beihang University, Beijing 100083, China
e-mail: anqiangh@163.com

D. Liu
Management Faculty, Chengdu University of Information Technology, Chengdu 610103, China
e-mail: 264885613@qq.com

S. Wang (✉)
Academy of Mathematics and Systems Science, Chinese Academy of Sciences, Beijing 100190, China
e-mail: sywang@amss.ac.cn

experimental results in two customer churn prediction datasets show that FSDTE can achieve better performance compared with the traditional churn prediction strategies, as well as three existing transfer learning strategies.

Keywords Customer churn prediction · Transfer ensemble model · Feature selection · GMDH-type neural network · Transfer learning

1 Introduction

Customer churn is defined as the propensity of customers to cease doing business with a company in a given period [1]. Because the cost to obtain a new customer is approximately 5–6 times more than that to retain an old customer [2], customer retention is very important for an enterprise in a fiercely competitive market. To support the enterprises and reduce churn rate, we need to scientifically identify the customers that are at high risk of churn and optimize marketing intervention resource to retain more customers. Therefore, customer churn prediction is one of the key steps to maximize the value of customers for the enterprise and enhance the core competency of the enterprise [3].

At present, the most commonly used methods for customer churn prediction include decision tree, artificial neural network, logistic regression, Bayesian classifier, and support vector machine (SVM), etc. [4–7]. In customer churn prediction, the class distribution of customer data is often imbalanced, that is, the number of churn customers constitutes only a very small minority of the data (usually 2 % of the total customers) [8]. When the class distribution of the data is imbalanced, the misclassification rate in the above classification models for churn customers is much higher than that of non-churn customers [9,10]. However, the value of accurate classification for a churn customer is often higher than that of a non-churn customer [11]. Therefore, efficient handling of the class imbalance issue is the key to successful customer churn prediction.

In general, there are two types of approaches to deal with the class imbalance issue in customer churn prediction: data-level and algorithm-level solutions. Data-level solutions mainly use resampling techniques, such as random over-sampling for the minority customers and random down-sampling for the majority customers, to balance the class distribution of the training set and construct the prediction model. In [12], the authors adopted over-sampling to handle the class imbalance issue in churn prediction. In [13], the authors combined the resampling technique with ensemble learning method to predict the churn. Algorithm-level solutions attempt to adapt existing classification algorithms to strengthen learning with regard to the minority class. Such solutions mainly introduce cost-sensitive learning technique and assign different misclassification costs to the customers from different classes. For instance, in [14], the authors utilized two cost-sensitive classification models, cost-sensitive decision tree and AdaCost, to predict the customer churn. The common characteristic of the two types of methods is that they only use the original information in the inner system (target domain) to handle the class imbalance issue and do not generate new information. According to the statistics learning theory, under certain sample information capacity, model accuracy has an upper limit [15]. Therefore, improving the prediction accuracy of the minority customers for both types of solutions is usually on the condition that the prediction accuracy of the majority customers is sacrificed.

A popular phenomenon exists in the real customer churn prediction. There are a large number of customer data in related source domains, which may be from different districts, businesses, periods, or from different enterprises in the same industry. Although the customer

data in the source and target domains are very similar, they are often subject to different distributions. Achieving satisfactory performance in this case is difficult for most of the traditional models, such as logistic regression and Bayesian classifier, because they are based on the assumption that the training data and the test data are subject to the same distribution [16]. Therefore, effective integrating the data from the source and target domains is important to improve the churn prediction performance with imbalanced class distribution.

The transfer learning proposed in machine learning area provides a new idea for this issue, and its main idea is to utilize the data of related source domain tasks to assist in modeling of target task [16]. In recent years, transfer learning has been applied to many areas such as text mining, image recognition, and so on. However, it is seldom applied to the customer churn prediction.

Combining the transfer learning, multiple classifiers ensemble (MCE) [17], and GMDH (group method of data handling)-type neural network [18,19], this study proposes a feature-selection-based dynamic transfer ensemble model (FSDTE) and applies it to customer churn prediction. The experimental results in two customer churn prediction datasets show that FSDTE can achieve better performance compared with the traditional churn prediction models, as well as some existing transfer learning models such as TFS, TrBagg, and TrAdaBoost.

The structure of this study is organized as follows: It simply introduces the related theories in Sect. 2, proposes the work principle and detailed steps of FSDTE in Sect. 3, and presents the experimental design and detailed results analysis in Sect. 4. Finally, the conclusions are included in Sect. 5.

2 Related theories

2.1 Multiple classifiers ensemble

Classification is one of the key technologies in data mining and has been applied to many areas such as speech recognition, text classification, and image processing. Many classification learning algorithms generate a single classifier (e.g., a decision tree or neural network) that can be used to predict the class labels of new patterns. However, because the data in real classification issues include much noise, it is difficult to classify accurately in the whole pattern space with single classifier [20]. If we can integrate the classification results of some classifiers with MCE technique, and each classifier plays role in its dominant area, then it is hopeful to improve the classification accuracy [21,22].

A successful MCE system should have the following two characteristics [23]: First, the base classifiers for ensemble system have higher classification accuracy, at least greater than 0.5 [21,22]; second, the classification results of base classifiers should be diverse. To ensure the diversity among the base classifiers, four different strategies are proposed [20]: (1) subsample the training set, such as Bagging strategy [24]—this method requires the execution of the training algorithm several times in different subsets of the training set; (2) manipulate the input features (i.e., select feature subsets) to train different classifiers—the representative method is random subspace method [25]; (3) manipulate the output classes—this technique involves the manipulation of the output (y) values that are presented to the learning algorithm; (4) diversify the algorithm parameters, which varies with the parameter values of the algorithm in the same dataset to achieve diverse results.

The constructing of classifier ensemble strategies is a key step in MCE. The existing ensemble strategies can be divided into two types: (1) static classifier ensemble (SCE), which selects a unified ensemble scheme for all test patterns; (2) dynamic classifier ensemble (DCE).

In fact, different test patterns usually have different classification difficulties. Intuitively, if we adopt different classifiers for different test patterns, the classification performance may be better than that by SCE. This is also the basic idea of DCE. Further, DCE strategies contain dynamic classifier selection (DCS) [22,26] and dynamic classifier ensemble selection (DCES) [27], in which the former selects a single best classifier for each test pattern, and the latter selects different ensemble solutions for different test patterns, noting that the FSDTE model proposed in this study belongs to DCS strategy.

2.2 Transfer learning theory

The traditional machine learning methods usually suppose that the training dataset and the test dataset are subject to the same distribution. In fact, the same distribution hypothesis may not be satisfied in many situations, because the distribution will change as the time goes on. Different from the traditional machine learning, the main idea of transfer learning is to utilize the data of related tasks to assist in modeling of target task [16] (see Fig. 1). It does not need the same distribution hypothesis, but utilizes the relationship between different tasks and the knowledge learned from one environment to assist in learning task in new environment.

In recent years, many scholars have focused on transfer learning strategies, and the representative researches are as follows: the feature-based transfer learning strategy TFS [28], the instance-based TrBagg strategy [29], and the instance-based TrAdaBoost strategy [30].

TFS [28] is based on a commonly used feature selection algorithm and supervised forward feature selection (SFFS). It first employs SFFS to select some features in target training set T_{rain} for composing the current best feature subset *cuf* and computes the *confidence* for each pattern in source domain S. Next, some patterns are randomly selected from S based on the *confidence* and are combined with T_{rain} for selecting *fn* features by SFFS. Then, with H loops of the above random pattern selection and feature selection processes, H feature subsets are selected, and the feature chosen with the highest frequency but not yet in *cuf* is added. The above two steps are repeated until the size of *cuf* reaches the predefined size or until the algorithm has been run with predefined iterations. Finally, it trains a classification model in the final feature subset *cuf*. Obviously, the time complexity of this strategy is very high.

TrBagg [29] is the expansion of the MCE strategy Bagging [24]. It includes two phases: training and filtering. In the training phase, data in source domain and target domain are combined to construct the training dataset. Next, N base classifiers $C = \{C_1, C_2, \ldots, C_N\}$

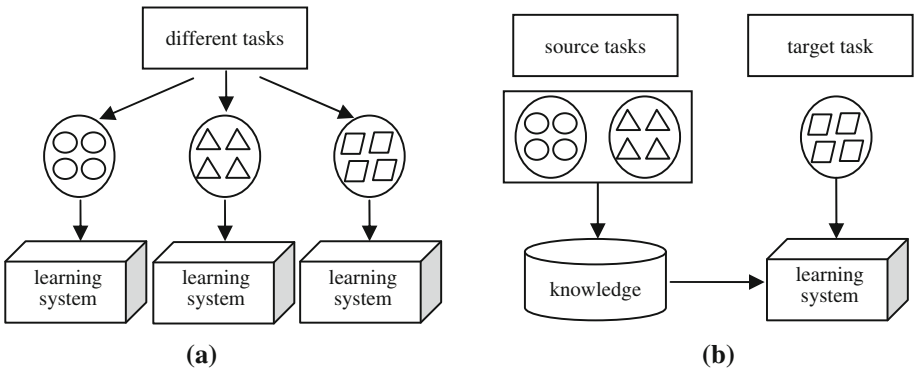

Fig. 1 The comparison between traditional machine learning and transfer learning. **a** Traditional machine learning. **b** Transfer learning

are trained similarly to the standard Bagging method. In the filtering phase, a subset of weak classifiers, $C^* \subseteq C$, is selected to minimize the empirical errors in the target set derived by majority voting of classifiers in C^*. For the detailed modeling process of TrBagg, the reader can refer to [29].

TrAdaBoost [30] is derived from the MCE strategy AdaBoost [31]. The basic idea of AdaBoost is that a training pattern is thought to be difficult one if it is misclassified by the trained model. Therefore, AdaBoost will increase the weight of the pattern to emphasize it and expect it to be classified correctly in the next iteration. In TrAdaBoost, AdaBoost is still applied to the target training set T_{rain}. However, the patterns in source domain are thought to be different from that in T_{rain} when they are misclassified, and Hedge (β) algorithm [31] is adopted to decrease their weights further.

2.3 GMDH-type neural network

GMDH-type neural network is a self-organizing modeling technique [19], and it can select some features that are mostly related to the research object from the whole feature space [32]. In recent years, the GMDH-type neural network has been applied in a broad range of areas such as engineer, science, and economics successfully [33–36].

Given a feature selection issue, the GMDH-type neural network builds the general relationship between the output and input variables in the form of mathematical description, which is also called reference function. Generally, the description can be considered as a discrete form of the Volterra functional series or Kolmogorov–Gabor polynomial:

$$\begin{aligned} Y &= f(X_1, X_2, \ldots, X_n) \\ &= a_0 + \sum_{i=1}^{n} a_i X_i + \sum_{i=1}^{n}\sum_{j=1}^{n} a_{ij} X_i X_j + \sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n} a_{ijk} X_i X_j X_k + \cdots, \end{aligned} \tag{1}$$

where Y is the output, $X = (X_1, X_2, \ldots, X_n)$ is the input vector, and a is the vector of coefficients or weights. Especially, the form of the first-order (linear) K–G polynomial including n variables (neurons) can be expressed as follows:

$$f(X_1, X_1, \ldots, X_n) = a_0 + a_1 X_1 + a_2 X_2 + \cdots + a_n X_n. \tag{2}$$

If the linear reference function like the form of Eq. (2) is chosen, the subsections of Eq. (2) are regarded as n initial models of the GMDH-type neural network, that is, $v_1 = a_1 X_1, \ldots, v_n = a_n X_n$. GMDH-type neural network has multi-layer structure. Before modeling, the training set is equally divided into model learning set A and model selecting set B at random. It starts from the initial model set composed by reference function, carries on parameter estimating by inner criterion (least squares) and gets middle candidate models (inherit, mutation) in set A, evaluates the middle candidate models by external criterion in set B, and chooses some best ones to get into the next layer, until it finds the optimal complexity model by the termination principle, which is presented by the optimal complexity theory: Along with the increase in model complexity, the value of external criterion will increase first and then decrease, and the global extreme value corresponds to the optimal complexity model [32]. The detailed modeling process is as follows.

As for n initial models, suppose that they are fed in pairs at each unit and the reference function is the first order, then $C_n^2 (= n(n-1)/2)$ candidate models with the form below are generated at the first layer (see Fig. 2):

$$w = f(v_i, v_j);\ i, j = 1, 2, \ldots, n;\ i \neq j, \tag{3}$$

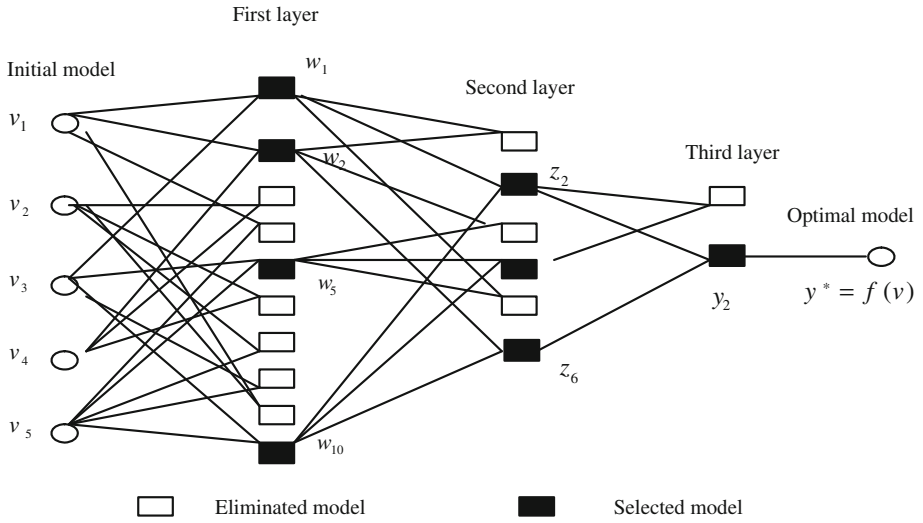

Fig. 2 The diagram of optimal model generation process by GMDH-type neural network

where $f()$ is partial function as in Eq. (2) and w is its estimated output. Then, the outputs of $Q_1(\ \leq\ C_n^2)$ functions are selected as per the external criterion value to pass on to the second layer as inputs in pairs. In the second layer, we check the forms of the functions

$$z = f(w_i, w_j);\ \ i, j = 1, 2, \ldots, Q_1;\ \ i \neq j. \tag{4}$$

The number of such functions is $C_{Q_1}^2$. Further, the outputs of $Q_2(\leq\ C_{Q_1}^2)$ functions are selected to pass on to the third layer. In the third layer, we estimate the form:

$$y = f(z_i, z_j);\ \ i, j = 1, 2, \ldots, Q_2;\ \ i \neq j. \tag{5}$$

The number of such functions is $C_{Q_2}^2$. The process continues and stops after finding the optimal complexity model by the optimal complexity theory. In this way, the algorithm can determine the input variables, structure, and parameters of final model automatically, accomplish the process of self-organizing modeling, and also can avoid over-fitting [37]. After finding y^*, we only need to reverse from the last layer to the input layer to find the features contained in y^*. It can be seen from Fig. 2 that v_1, v_3, v_4, and v_5 are selected, i.e., the second feature X_2 is excluded in the final feature subset.

3 Feature-selection-based dynamic transfer ensemble model

3.1 The basic idea of FSDTE

The FSDTE model aims to transfer some useful information from the related source domain to assist in modeling for target domain. As for a classification problem, suppose that the

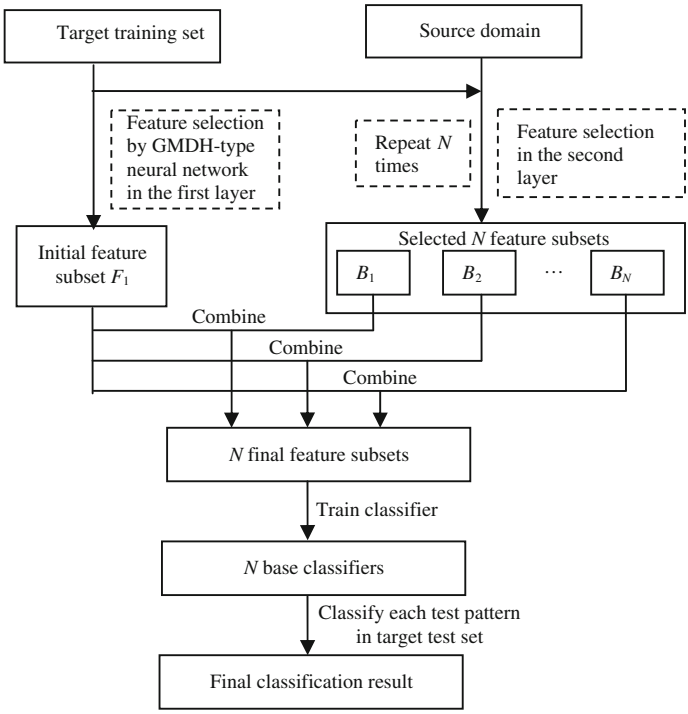

Fig. 3 The flowchart of FSDTE model

target domain T and the source domain S contain m_1 and m_2 patterns, respectively, and they are subject to different distributions. Further, T is divided into two subsets: target training set T_{rain} and target test set T_{est}. And there are m_3 patterns in T_{rain}. Meanwhile, let the source domain and target domain have the same feature space with n features, that is, $F = \{f_1, f_2, \ldots, f_n\}$.

The FSDTE model mainly conducts a two-layer feature selection integrating pattern selection based on both the target and source domain data. In the first layer, an initial subset F_1 from the whole feature space F is selected by GMDH-type neural network only in the target training set T_{rain}. In the second layer, the most appropriate patterns are selected from the source domain and integrated into T_{rain}, and the MI between the remaining features and the class variable is calculated. Then, some features with higher MI are selected and combined with F_1 to construct a new feature subset. Repeating N times in the second layer, N new feature subsets are obtained, and then, a base classifier is trained in each subset. Finally, we classify the test patterns in target test set by the trained base classifiers. For each test pattern $x_j^* \in T_{\text{est}}$, the FSDTE model first constructs its local area R_K and then finds the base classifier with the highest classification accuracy in R_K and classifies x_j^* just by this classifier to obtain the final classification result of x_j^*. The flowchart of the FSDTE model is shown in Fig. 3.

In the second layer, to avoid negative transfer to some extent, the concept of *confidence* [28] is introduced. For each pattern $x_i (i = 1, 2, \ldots, m_2)$ in source domain, it finds K nearest neighbors from target training set T_{rain} for x_i (the Euclidean distance is regarded as the distance measure). Then, the *confidence* of x_i can be defined:

$$C_i = 1 - \frac{1}{K} \sum_{(x_s^j, y_s^j) \in \Omega_i} \left| y_s^j - y^i \right|, \tag{6}$$

where Ω_i is the set of K nearest neighbors of x_i, y_s^j is the class label of pattern in Ω_i, $and y^i$ is the class label of x_i. The higher the *confidence* is, the larger the selection probability of the pattern is. Noting that the class distribution of the source domain S is usually highly imbalanced, if we select the patterns with higher *confidence* from S directly, all the selected patterns may be non-churn customers, which cannot improve the prediction accuracy of churn customers. Therefore, we divide the source domain into two subsets: churn customer set S_1 with t_1 patterns and non-churn customer set S_2 with t_2 patterns. Then, we calculate the sampled probabilities $b^q = (b_1^q, b_2^q, \ldots, b_{t_q}^q)(q = 1, 2)$ of all patterns in the two subsets, which is defined as follows:

$$b_j^q = \frac{Cj}{\sum_{k=1}^{t_q} C_k}, j = 1, 2, \ldots, t_q; q = 1, 2, \tag{7}$$

where b_j^q is the sampled probability of the jth pattern in the qth subset S_q, and C_j is the *confidence* of the jth pattern obtained by Eq. (6). Further, r percent patterns with higher *confidence* are selected from the subsets S_1 and S_2, respectively, according to the sampled probabilities and combined with the target training set T_{rain} to construct a new set T'_{rain}.

The feature selection in the second layer is implemented by calculating MI in T'_{rain} between the class variable and each feature in the remaining feature set $(F - F_1)$:

$$I_t(Y, f_t) = \sum_{Y, f_t} P(Y, f_t) \log \frac{P(Y, f_t)}{P(Y)P(f_t)}, t = 1, 2, \ldots, |F - F_1|, \tag{8}$$

where Y is the class variable, f_t is the tth feature in $F - F_1$, $|F - F_1|$ is the number of features in $F - F_1$, $P(Y, f_t)$ is the joint probabilities, and $P(Y)$ and $P(f_t)$ are the probabilities of Y and f_t, respectively. The larger the value of MI is, the stronger the dependence between the two variables is. After obtaining the MI between each feature and class variable, we calculate the selected probability g_t of each feature in $F - F_1$ as follows:

$$g_t = \frac{I_t(Y, f_t)}{\sum_{k=1}^{|F-F_1|} I_k(Y, f_k)}, t = 1, 2, \ldots, |F - F_1|. \tag{9}$$

The larger the value of g_t is, the greater the selected probability of the corresponding feature f_t is. Finally, p percent features are selected from $F - F_1$ according to the selected probabilities and combined with the initial subset F_1 to construct a new feature subset for training a base classifier. In general, the classification techniques typically perform best when the class distribution is approximately even [11]. Therefore, the final training subset is over-sampled to approximate an even class distribution.

3.2 Algorithm description

In general, the FSDTE model includes three phases: (1) feature selection by GMDH-type neural network in the first layer; (2) feature selection in the second layer; (3) classification for each test pattern in target test set. The pseudo-code of FSDTE model is as follows:

Phase I: Feature selection by GMDH-type neural network in the first layer

1. Divide the target training set T_{rain} into two parts: model learning set A and model selecting set B, and set model layer $L = 1$, $Q_0 = n$, and the smallest external criterion value $M = e$ (e is a large positive number, such as $e = 10000$);

2. Combine every two initial models to obtain C_n^2 middle candidate models in the first layer (see Fig. 2), estimate the model parameters by least-square (LS) in set A, and then compute the external criterion (the GMDH-type neural network has an external criterion system, and the mean-squared error is selected in this study) value for each model in set B;
3. Select $Q_L (\leq Q_{L-1})$ models with smaller external criterion values to enter the next layer, consider the smallest external criterion value $M_{\min}$ in this layer, and if $M_{\min} \geq M$, then STOP, and the optimal complexity model y^* is the one with smallest external criterion value in the previous layer (the features in y^* just construct the feature subset F_1), else set $M = M_{\min}$ and CONTINUE;
4. Repeat Steps 2–3 with $L = L + 1$;

Phase II: Feature selection in the second layer

5. Divide the source domain S into subsets S_1 and S_2 according to the class label and calculate the sampled probability of each pattern by Eq. (7);
6. Select r percent patterns from S_1 and S_2, respectively, according to the sampled probabilities, combine these patterns with T_{rain} to construct a new set T'_{rain}, and compute the selected probability of each feature in the remaining feature set $F - F_1$ by Eq. (9);
7. Select p percent features from $F - F_1$ according to the selected probabilities and combine them with $F_1$to compose a new feature subset F'_1, get a training subset according to F'_1 by mapping in T'_{rain} and balance it with over-sampling, and then train a base classifier in the balanced training set;
8. Repeat Steps 6–7 N times, and N base classifiers are obtained;

Phase III: Classification for each test pattern

9. For each test pattern $x_j^* \in T_{\text{est}}$,
   9.1 Find its K nearest neighbors from T_{rain} to construct a local area R_K;
   9.2 Classify R_K with N trained base classifiers, respectively;
   9.3 Find C^* with the highest classification accuracy in R_K and classify x_j^* by C^*.

4 Empirical analysis

In order to analyze the prediction performance of FSDTE proposed in this study, we experimented in two customer churn prediction datasets. Meanwhile, we compared FSDTE model with the following strategies: (1) traditional Bagging [24] by utilizing all data (Bagging), which trains N classifiers by uniting the data in the source domain with those in the target domain without distinction; (2) traditional Bagging by utilizing target domain data only (Bagg-OT), which trains N classifiers by using the data in the target domain; (3) transfer feature selection method TFS [28]; (4) instance-based transfer learning strategy TrBagg [29]; and (5) instance-based transfer learning strategy TrAdaBoost [30].

4.1 Datasets description

(1) The "churn" dataset

The "churn" dataset is from machine learning UCI database in California University [38]. It deals with cellular service provider's customers and the data pertinent to the voice calls they make. Customers have a choice of service providers, or companies providing them with cellular network services. When these customers change cellular service provider, they are said to churn which results in a loss of revenue for the previous cellular service provider.

Table 1 Attribute description of "churn" dataset

Feature	Name	Feature	Name
X_1	State	X_{10}	Total evening calls
X_2	Account length	X_{11}	Total evening charge
X_3	International plan	X_{12}	Total night minutes
X_4	Voice mail plan	X_{13}	Total night calls
X_5	Number of voice mail messages	X_{14}	Total night charge
X_6	Total day minutes	X_{15}	Total international minutes
X_7	Total day calls	X_{16}	Total international calls
X_8	Total day charge	X_{17}	Total international charge
X_9	Total evening minutes	X_{18}	Number of calls to customer service

There are 3,333 patterns, among which 2,850 belong to non-churn customers and 483 belong to churn customers; The ratio is 5.9006, and the class distribution is highly imbalanced.

The dataset includes 20 features. Two features, namely, phone number (unique for each subscriber) and area code, are deemed irrelevant. Therefore, the remaining 18 features form the set of decision criteria for churn prediction (Table 1).

To conduct transfer learning, we need to partition this dataset into target domain and source domain. According to the first feature X_1 (*state*), the customers come from 50 states and District of Columbia of the United States. The data from different districts may be subject to different distributions. At the same time, the number of customers in each district is very small and the largest number is 106 (in West Virginia). If we only regard one district as the target domain, the churn customers will be very few, and the experimental results may be instable. Thus, to ensure that more than 10 churn customers are included in the target test set and no loss of generality transpires, we sorted the values of the variable *state* from A to Z and selected 251 customers from the first four states (Alaska, Alabama, Arkansas, and Arizona) as the elements of target domain T, and the remaining customer data composed of the source domain S.

(2) The "China-churn" dataset

This dataset is from a credit card business in one commercial bank in Chongqing, China ("China-churn"). The data interval is from May 2010 to December 2010. According to the basic principles of selecting the index of customer churn prediction and the convenience of obtaining index, we selected 25 churn variables (Table 2), among which 8 are continuous variables and 17 are discrete variables. For the class variable, we defined the customers who logged out their cards between May and December or did not spend in 3 continuous months as the churn customers. After simple data cleaning, we obtained 1,255 patterns in target domain, among which 1,151 are non-churn customers and 104 are churn customers; the customer churn rate is 8.29 %, and it is highly class imbalanced data. Because there are only a few churn customer patterns, it is hard to achieve satisfactory performance by the model trained in the target domain data directly. However, it was lucky that we obtained the dataset of another credit card business from the customer system of the same bank, among which 1802 are non-churn customers and 198 are churn customers, and they construct the source domain.

To determine whether the distributions of the source domain and target domain are different, we introduced the multivariate two-sample testing procedure proposed in [39]. It can be roughly divided into the following steps: (1) create a predictor variable training set

Table 2 Attribute description of "China-churn" dataset

Feature	Name	Feature	Name
X_1	Total consumption times	X_{14}	Cash times in the last 1 month
X_2	Total consumption amount	X_{15}	Cash times in the last 2 months
X_3	Total cash times	X_{16}	Cash times in the last 3 months
X_4	Customer survival time	X_{17}	Cash times in the last 6 months
X_5	Total contributions	X_{18}	Months of transaction times reducing continuously
X_6	Valid survival time	X_{19}	If overdue in the last 1 month
X_7	Average amount ratio	X_{20}	If overdue in the last 2 months
X_8	Whether associated charge	X_{21}	Amount usage ratio in the last 1 month/ historical average usage ratio
X_9	Consumption times in the last 1 month		
X_{10}	Consumption times in the last 2 months	X_{22}	Sex
X_{11}	Consumption times in the last 3 months	X_{23}	Annual income
X_{12}	Consumption times in the last 4 months	X_{24}	Nature of work industry
X_{13}	Consumption times in the last 5 months	X_{25}	Education

$\{u_i\}_1^{m_1+m_2} = \{t_i\}_1^{m_1} \cup \{s_i\}_1^{m_2}$ by pooling the two samples, i.e., the target domain T and the source domain S, and assign a response value $y_i = 1(1 \leq i \leq m_1)$ to the observations originated from the first sample while assign $y_i = -1(m_1 + 1 \leq i \leq m_1 + m_2)$ to those from the second sample; (2) a binary classification learning machine (e.g., the support vector machine is selected in this study) is applied to this training data to produce a scoring function $L_m(u)$, and then, this function is used to score each observation $\{score_i = L_m(u_i)\}_1^{m_1+m_2}$; (3) generate two sets of score values $Score_+ = \{score_i)\}_1^{m_1}$ and $Score_- = \{score_i)\}_{m_1+1}^{m_1+m_2}$, regard the sets of numbers $Score_\pm$ as a random sample from respective probability distributions with densities $p_+(score)$ and $p_-(score)$, apply a *univariate* two-sample test (e.g., the two independent samples t test is introduced in this study) for the equality of these densities $p_+(score) = p_-(score)$, and compute the test statistic $\hat{t}$; (4) let $\{j(i)\}_1^{m_1+m_2}$ u represent a random permutation of the integers $\{i\}_1^{m_1+m_2}$ and construct a dataset $\left\{y_{j(i)}, u_i\right\}_1^{m_1+m_2}$ in which the actual response values $\{y_i\}_1^{m_1+m_2}$ are randomly permuted among the predictors $\{u_i\}_1^{m_1+m_2}$; (5) train a support vector machine by these data, score the observations and compute the t test statistic $\hat{t}_1'$; (6) repeat Steps 4–5 1,000 times to generate a set of test statistic values $\left\{\hat{t}_i'\right\}_1^{1000}$, sort them in ascending order according to their absolute values; (7) giving a significance level α, one can reject the null hypothesis $p_+(score) = p_-(score)$ if $|\hat{t}| > |\hat{t}'_{1000*(1-\alpha)}|$. In this study, we let $\alpha = 0.05$, and the test results are shown in Table 3. It can be seen that there is significant difference between the distributions of the target domain and source domain in both datasets.

4.2 Experimental setup

Before training the models, we need to partition the target domain T into target training set T_{rain} and target test set T_{est}. In this study, we adopted the random sampling without replacement method to select 30 % patterns from T to construct T_{est}, and the remaining patterns composed of T_{rain}.

Table 3 Test results of multivariate two-sample testing in two datasets

Datasets	$\lvert\hat{t}\rvert$	$\lvert\hat{t}^{\gamma}_{950}\rvert$
Churn	74.6221	2.0088
China-churn	54.1384	11.5816

Many classification algorithms can be used to generate base classifiers; in this study, we choose support vector machine (SVM) [40] for its popularity and immense success in various customer classification tasks. When training SVM, the choice of kernel function is very important. We found that the classifier based on radial basis kernel (RBK) could obtain the best performance through experimental comparison; thus, we designated it as the kernel function of SVM. The kernel parameter of the RBK and the regularization parameter were set as the default values. We did not optimize the parameters of the SVMs because our concern was more on the relative performance of the compared ensemble models, rather than on their absolute performance.

It is worth noting that Bagging, Bagg-OT, TFS, TrBagg, and TrAdaBoost models all belong to static ensemble models. In [41], the authors found that the static ensemble models usually could achieve their best performance when the number of base classifiers for ensemble equaled 50. Therefore, we let the size of base classifier pool for the five models be 50. For the other parameters in TFS, TrBagg, and TrAdaBoost models, we let them be the values which make the models perform best by repeated experiments. Meanwhile, no one among the five models considers the impact of class imbalance on performance. To ensure the fairness of comparison, we balanced the class distribution of data by using the over-sampling technique before training the base classifiers. In addition, all experiments were performed on the MATLAB 6.5 platform with a dual-processor 2.1 GHz Pentium 4 Windows computer. For each model, the final classification result was the average of the results from 10 iterations of the experiment.

4.3 Evaluation criteria

To evaluate the performance of the strategies referred in this study, we introduced the confusion matrix in Table 4. The following three commonly used evaluation criteria were adopted [42,43]:

(1) The area under the receiver operating characteristic curve (AUC): The receiver operating characteristic (ROC) curve is an important evaluation criterion of classification model in the data with imbalanced class distribution. For an issue of two classes, the ROC graph is a true-positive rate–false-positive rate graph, where Y-axis is true-positive rate ($\mathrm{TP}/(\mathrm{TP}+\mathrm{FN}) \times 100\,\%$) and X-axis is false-positive rate ($\mathrm{FP}/(\mathrm{FP}+\mathrm{TN}) \times 100\,\%$). However, sometimes, it is difficult to compare ROC curves of different models directly, so AUC is more convenient and popular;

(2) Type I accuracy = $\frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FN}}$;

(3) Type II accuracy = $\frac{TN}{FP+TN}$.

4.4 Impact of feature selection, pattern selection, and over-sampling

Feature selection and pattern selection are two main components of the proposed FSDTE model. In addition, over-sampling is adopted to balance the class distribution of the training set. To assess the impacts of feature selection, pattern selection, and over-sampling on

Table 4 Confusion matrix

	Predicted positive	Predicted negative
Actual positive (churn customer)	TP (the number of True Positives)	FN (the number of False Negatives)
Actual negative (non-churn customer)	FP (the number of False Positives)	TN (the number of True Negatives)

Table 5 The experimental results in two datasets

	AUC ± SD	Type I accuracy ± SD	Type II accuracy ± SD
The results in "churn" dataset			
FSDTE	0.8691 ± 0.0212	0.8081 ± 0.0312	0.7853 ± 0.0362
FSDTE1	0.8210 ± 0.0345	0.6760 ± 0.0457	0.8227 ± 0.0451
FSDTE2	0.8428 ± 0.0283	0.7096 ± 0.0651	0.8089 ± 0.0413
FSDTE3	0.7893 ± 0.0476	0.6398 ± 0.0721	0.8161 ± 0.0305
The results in "China-churn" dataset			
FSDTE	0.9723 ± 0.0166	0.9350 ± 0.0557	0.9208 ± 0.0194
FSDTE1	0.9310 ± 0.0300	0.8160 ± 0.0617	0.9170 ± 0.0135
FSDTE2	0.9483 ± 0.0187	0.8976 ± 0.0775	0.9089 ± 0.0253
FSDTE3	0.9048 ± 0.0244	0.8198 ± 0.0857	0.8861 ± 0.0307

the performance of FSDTE model, we experimented with the following four strategies: (1) FSDTE; (2) FSDTE without over-sampling, which is similar to FSDTE except that it does not balance the training set with over-sampling (called FSDTE1); (3) FSDTE without feature selection, which selects r percent patterns with higher *confidence* randomly from the source domain every time, combines them with the target training set to obtain a new training set, and then balances it with over-sampling to train a classifier (called FSDTE2); and (4) FSDTE without pattern selection, which only utilizes the target training set T_{rain}, conducts the two-layer feature selection similar to the FSDTE model in T_{rain}, derives a training subset by mapping each time, and balances the subset with over-sampling to train a classifier (called FSDTE3).

The proposed FSDTE model has four parameters: the number of nearest neighbors K, number of base classifiers N, r percent patterns selected from the source domain S each time, and p percent features selected from the remaining feature subset $F - F_1$. In this section, we let $K = 5$, $N = 20$, $r = 70$, and $p = 70$. The average value of the results of 10 experiment runs and standard deviation (s.d.) of each evaluation criterion of the four strategies are shown in Table 5. The FSDTE model can always achieve the largest AUC values in the two datasets, after which come FSDTE2, FSDTE1, and finally FSDTE3. Therefore, the impact of pattern selection on the performance of FSDTE model is the largest, followed by those of over-sampling and feature selection, and these results also demonstrate that it is very important to transfer some suitable patterns from the source domain to target domain.

However, the results above do not mean that feature selection is not important because it can effectively eliminate the redundant features and generate diverse base classifiers. Take the "China-churn" dataset as an example, Table 6 shows the selected features in the two layers of the FSDTE model. The last column in Table 6 displays the number of different

Table 6 Selected features in "China-churn" dataset

Feature subsets	Features selected in the first layer	Features selected in the second layer	Number of different features compared with 1st subset
1		$X_2, X_9, X_{12}, X_{13}, X_{15}, X_{16}, X_{17}, X_{18}, X_{19}, X_{20}, X_{21}, X_{25}$	–
2		$X_8, X_{10}, X_{11}, X_{12}, X_{15}, X_{16}, X_{17}, X_{18}, X_{19}, X_{20}, X_{21}, X_{25}$	3
3		$X_8, X_9, X_{10}, X_{11}, X_{12}, X_{13}, X_{15}, X_{16}, X_{17}, X_{20}, X_{22}, X_{25}$	4
4		$X_8, X_9, X_{10}, X_{11}, X_{12}, X_{15}, X_{16}, X_{17}, X_{18}, X_{19}, X_{20}, X_{25}$	3
5		$X_8, X_{10}, X_{12}, X_{14}, X_{15}, X_{16}, X_{17}, X_{18}, X_{19}, X_{20}, X_{21}, X_{25}$	3
6		$X_8, X_9, X_{11}, X_{12}, X_{13}, X_{15}, X_{16}, X_{17}, X_{18}, X_{19}, X_{20}, X_{25}$	2
7		$X_2, X_9, X_{13}, X_{14}, X_{15}, X_{16}, X_{17}, X_{18}, X_{19}, X_{21}, X_{22}, X_{25}$	2
8		$X_2, X_9, X_{11}, X_{12}, X_{14}, X_{15}, X_{16}, X_{17}, X_{18}, X_{20}, X_{21}, X_{25}$	2
9	$X_1, X_3, X_4, X_5, X_6, X_7, X_{23}, X_{24}$	$X_8, X_{10}, X_{11}, X_{13}, X_{15}, X_{16}, X_{17}, X_{18}, X_{19}, X_{20}, X_{21}, X_{25}$	3
10		$X_2, X_8, X_9, X_{10}, X_{12}, X_{15}, X_{16}, X_{17}, X_{19}, X_{20}, X_{21}, X_{25}$	2
11		$X_2, X_{11}, X_{12}, X_{14}, X_{15}, X_{16}, X_{17}, X_{18}, X_{19}, X_{20}, X_{22}, X_{25}$	3
12		$X_2, X_9, X_{10}, X_{11}, X_{12}, X_{13}, X_{15}, X_{16}, X_{17}, X_{20}, X_{21}, X_{25}$	2
13		$X_2, X_8, X_{11}, X_{12}, X_{13}, X_{15}, X_{16}, X_{17}, X_{18}, X_{19}, X_{21}, X_{25}$	2
14		$X_2, X_{10}, X_{12}, X_{15}, X_{16}, X_{17}, X_{18}, X_{19}, X_{20}, X_{21}, X_{22}, X_{25}$	2
15		$X_2, X_8, X_9, X_{10}, X_{12}, X_{14}, X_{15}, X_{16}, X_{17}, X_{18}, X_{21}, X_{25}$	3
16		$X_2, X_8, X_{12}, X_{13}, X_{14}, X_{16}, X_{17}, X_{18}, X_{19}, X_{20}, X_{21}, X_{25}$	2
17		$X_2, X_9, X_{11}, X_{12}, X_{14}, X_{15}, X_{16}, X_{17}, X_{19}, X_{20}, X_{22}, X_{25}$	2
18		$X_2, X_8, X_9, X_{11}, X_{12}, X_{13}, X_{15}, X_{17}, X_{18}, X_{19}, X_{21}, X_{25}$	3
19		$X_8, X_{11}, X_{12}, X_{13}, X_{14}, X_{15}, X_{16}, X_{17}, X_{18}, X_{19}, X_{20}, X_{25}$	3
20		$X_8, X_9, X_{10}, X_{11}, X_{12}, X_{13}, X_{15}, X_{16}, X_{17}, X_{19}, X_{21}, X_{25}$	3

features between the first feature subset and the other ones. It can be seen that there are at least 2 different features between each pair of feature subsets. In fact, in FSDTE model, the patterns in N subsets T'_{rain} generated by N iterations are also different, which leads to different patterns in N final training subsets. Therefore, different feature subsets and different patterns in N iterations ensure the diversity among the trained base classifiers to a large extent, and further ensure the classification performance of the FSDTE model.

4.5 Sensitivity analysis of parameters

We investigated the robustness of the FSDTE model by examining how uncertainties in the four input parameters affect the model performance. We studied the impact of changes in parameter K on the churn prediction performance by letting the values of r, p, and N be 70, 70, and 20, respectively. Subsequently, we examined the sensitivity of parameter r and analyzed how the changes in p affect the classification performance. Finally, we conducted the sensitivity analysis of parameter N.

4.5.1 Changes in the number of nearest neighbors

In FSDTE, K represents the number of nearest neighbors in the local area. In the work [22], the authors proposed a dynamic classifier selection based on local accuracy (DCS-LA) and found that DCS-LA could achieve satisfactory classification performance when $K = 10$. However,

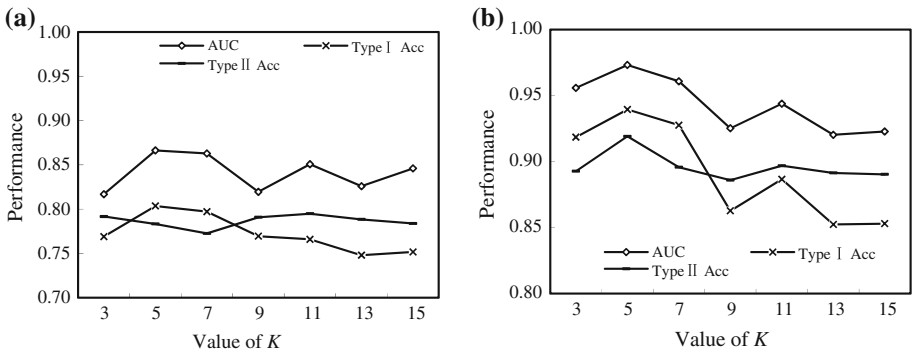

Fig. 4 The impact of parameter K on the performance. **a** Performance in "churn" dataset. **b** Performance in "China-churn" dataset

the optimal value of K may be different for different ensemble strategies. Therefore, to find a satisfactory value of K for FSDTE, we experimented with seven different values of K: 3, 5, 7, 9, 11, 13, and 15. The experimental results are shown in Fig. 4.

Figure 4a shows the performance in the "churn" dataset. The values of the three evaluation criteria of FSDTE model with different values of K show fluctuations, and the fluctuations of the values of Type I accuracy and AUC are the most intense in the "churn" dataset. In customer churn prediction issues, Type I accuracy (accuracy of the churn customers) is one of the criteria that concerns us most. The correct prediction of the churn customers can guide the decision maker in timely implementation of customer retention and in maximization of the enterprise's profits. It can be seen that the Type I accuracy reaches its maximum when $K = 5$. Further, the AUC, which is often regarded as the evaluation of the whole performance, reaches the maximum at this time as well. Therefore, the FSDTE model with $K = 5$ shows the best churn prediction performance in the "churn" dataset.

Figure 4b shows the performance in the "China-churn" dataset. When $K = 5$, the values of Type I accuracy and AUC reach their maxima, and then decrease with the increase in the value of K. Note that, the values of Type I accuracy are larger than those of Type II accuracy when $K = 3, 5$, and 7. In the class imbalanced churn prediction, the Type I accuracy is often lower than the Type II accuracy for most traditional models. The experimental results in the "China-churn" dataset indicate that in some cases, the FSDTE model can overcome the disadvantage of the traditional methods.

Based on the analysis of the two datasets, we conclude that the FSDTE model shows the best whole prediction performance when $K = 5$. Thus, in the following experiments of this study, we let $K = 5$.

4.5.2 Changes in the value of parameter r

The parameter r indicates the percent of patterns transferred from the source domain S. Figure 5 shows the customer churn prediction performance when the value of r varies from 10 to 100. Some fluctuations exist among the performance of the FSDTE model with different values of r, and the general trend of the performance increases first and then decreases in both datasets. The results seem reasonable. The FSDTE model selects r percent patterns from source domain S according to their similarity with the target training data to assist in modeling. When the value of r is relatively small, only the patterns with the highest similarity

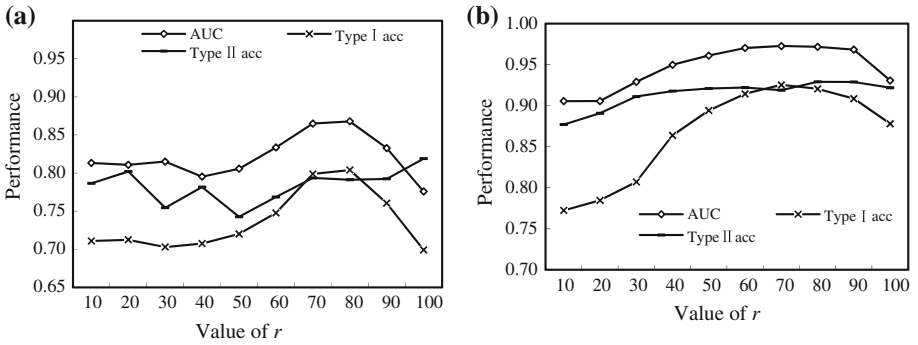

Fig. 5 The impact of parameter r on the performance. **a** Performance in "churn" dataset. **b** Performance in "China-churn" dataset

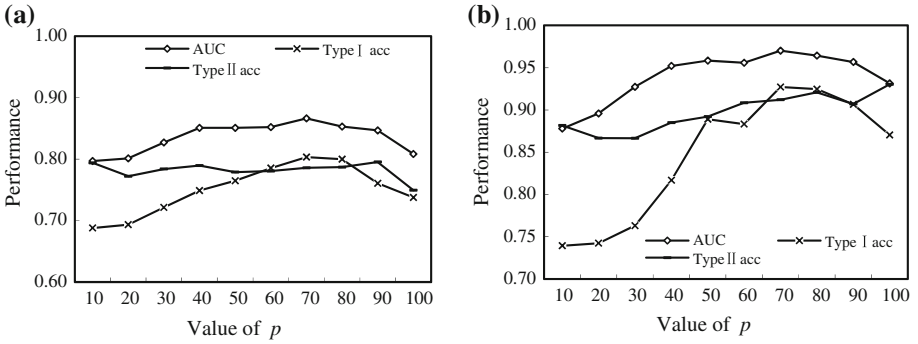

Fig. 6 The impact of parameter p on the performance. **a** Performance in "churn" dataset. **b** Performance in "China-churn" dataset

are transferred. In this case, the transfer may be positive. Therefore, the overall performance of the FSDTE model will increase first gradually when the value of r increases. However, the performance of the FSDTE model will not increase further when r becomes sufficiently large because it may transfer the patterns with high noise and damage the churn prediction performance, thereby resulting in negative transfer. The performance of the FSDTE model is comparable when $r = 70$ and 80 in both datasets. Therefore, in the following experiments of this study, we let $r = 70$ roughly.

4.5.3 Changes in the value of parameter p

The parameter p represents the percent of features selected from the remaining feature subset $F - F_1$ each time. The model performance when the value of p varies from 10 to 100 is shown in Fig. 6. The trend of the FSDTE model's performance is roughly the same as the case of the value changes of the parameter r. The FSDTE model with $p = 70$ shows the best performance in the "churn" dataset because, in this case, the Type I accuracy and AUC values reach their maxima. At the same time, the performance of the FSDTE model with $p = 70$ is also the best in the "China-churn" dataset. Hence, we let $p = 70$ in the following experiments.

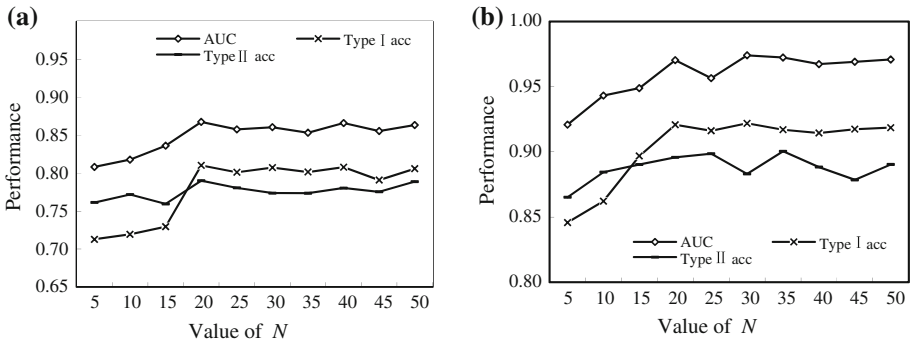

Fig. 7 The impact of parameter N on the performance. **a** Performance in "churn" dataset. **b** Performance in "China-churn" dataset

Table 7 Customer churn prediction performance of six models in "churn" dataset

Criteria	FSDTE	Bagging	Bagg-OT	TFS	TrBagg	TrAdaBoost
AUC	**0.8691**	0.8313	0.7676	0.8072	0.8503	0.7930
Type I accuracy	**0.8081**	0.6833	0.5021	0.5569	0.6615	0.4308
Type II accuracy	0.7853	0.7731	0.8282	0.8349	0.7822	**0.8890**

The boldface in Table 7 shows the maximum of each row

4.5.4 Changes in the number of base classifiers

We experimented with ten different values of N: 5, 10, 15, 20, 25, 30, 35, 40, 45, and 50. Figure 7 shows the model performance in the two datasets. The performance of the FSDTE model is gradually improved with the increase of N, reaches the maximum when $N = 20$, and then exhibits slight fluctuations with the further increase of Nin both datasets. The results are consistent with those presented in [41], which shows that the performance of dynamic classifier ensemble model nearly reaches its maximum when the number of base classifiers is within 20–25. Therefore, in the following experiments, we let $N = 20$.

4.6 Performance comparison with other models

Tables 7 and 8 show the values of three criteria for the six models in the two datasets. Through accuracy and AUC, we can determine the most and least advantageous models. However, the difference between these models is not clearly defined. Therefore, we conducted the McNemar's test [44] to determine whether the proposed FSDTE model can significantly outperform the other five models. The results of the McNemar's test are shown in Tables 9 and 10.

According to the experimental results above, we offer the following conclusions:

(1) In the "churn" dataset, the prediction performance of the proposed FSDTE model is significantly better than that of Bagging, Bagg-OT, TFS, TrBagg, and TrAdaBoost at 1 % statistical significance level. The performance of the Bagging model is significantly better than that of Bagg-OT, TFS, and TrAdaBoost at 1 % statistical significance level,

Table 8 Customer churn prediction performance of six models in "China-churn" dataset

Criteria	FSDTE	Bagging	Bagg-OT	TFS	TrBagg	TrAdaBoost
AUC	**0.9723**	0.9598	0.8791	0.9021	0.9594	0.9548
Type I accuracy	**0.9350**	0.8809	0.8047	0.8357	0.8689	0.8532
Type II accuracy	0.9208	0.9145	0.8981	0.9064	0.9139	**0.9350**

The boldface in Table 8 shows the maximum of each row

Table 9 McNemar's test for pairwise comparison of performance in "churn" dataset

Model	Bagging	Bagg-OT	TFS	TrBagg	TrAdaBoost
FSDTE	10.240 (0.0014)	12.893 (0.0003)	11.115 (0.0009)	7.0420 (0.0080)	12.000 (0.0005)
Bagging		10.321 (0.0013)	7.5000 (0.0062)	6.3230 (0.0119)	9.0310 (0.0027)
Bagg-OT			7.7580 (0.0050)	11.172 (0.0008)	6.5000 (0.0108)
TFS				9.4810 (0.0021)	3.3750 (0.0662)
TrBagg					10.321 (0.0013)

The results listed in Table 9 are the chi-squared values, and p values are in brackets

Table 10 McNemar's test for pairwise comparison of performance in "China-churn" dataset

Model	Bagging	Bagg-OT	TFS	TrBagg	TrAdaBoost
FSDTE	4.7800 (0.0280)	14.754 (0.0001)	11.161 (0.0008)	4.9830 (0.0256)	5.4920 (0.0191)
Bagging		13.043 (0.0003)	9.1430 (0.0025)	1.9290 (0.1649)	2.5210 (0.1124)
Bagg-OT			7.6030 (0.0058)	12.062 (0.0005)	11.391 (0.0007)
TFS				8.8170 (0.0030)	8.2030 (0.0042)
TrBagg					2.2070 (0.1374)

The results listed in Table 10 are the chi-squared values, and p values are in brackets

and significantly poorer than that of TrBagg at 5 % statistical significance level. The performance of the Bagg-OT model is significantly poorer than that of TFS and TrBagg at 1 % statistical significance level. Similarly, its performance is significantly poorer than that of TrAdaBoost at 5 % statistical significance level. The performance of the TFS model is significantly poorer than that of TrBagg at 1 % statistical significance level and significantly better than that of TrAdaBoost at 10 % statistical significance level. The TrBagg model outperforms TrAdaBoost at 1 % statistical significance level.

(2) In the "China-churn" dataset, the FSDTE model outperforms Bagg-OT and TFS at 1 % statistical significance level, and outperforms Bagging, TrBagg, and TrAdaBoost at 5 % statistical significance level. The Bagging model performs better than Bagg-OT and TFS at 1 % statistical significance level, and better than TrAdaBoost at 10 % statistical significance level. However, we cannot determine whether the Bagging outperforms TrBagg and TrAdaBoost at 10 % statistical significance level. The performance of the Bagg-OT model is significantly poorer than that of TFS, TrBagg, and TrAdaBoost at 1 % statistical significance level. The performance of the TFS model is significantly poorer than that of TrBagg and TrAdaBoost at 1 % statistical significance level. The performance of the TrBagg model has no significant difference compared with that of TrAdaBoost at 10 % statistical level.

(3) Among the six models, only the Type I accuracy of the FSDTE model is higher than its Type II accuracy in both datasets, which demonstrates that the FSDTE model can better deal with the imbalance class distribution issue in the customer churn prediction from one perspective.
(4) Bagging outperforms Bagg-OT in the two datasets, which demonstrates that the utilization of the customer data in the source domain can significantly improve the churn prediction performance.
(5) TrBagg and TrAdaBoost show comparable performance in the "China-churn" dataset, whereas the performance of TrAdaBoost is the lowest in the "churn" dataset. Thus, TrBagg shows more stable performance than TrAdaBoost. Finally, the customer churn prediction performance of TFS is poor in both datasets.

5 Conclusions

Customer churn prediction is an important concern for numerous domestic and global industries. This study combines transfer learning and multiple-classifier ensemble with GMDH-type neural network and proposes FSDTE for customer churn prediction. Unlike the traditional research paradigm in customer churn prediction, which only utilizes the customer data in target domain, FSDTE not only uses the data in target domain, but also utilizes the data in related source domains to assist in modeling. The experimental results in two customer churn prediction datasets show that FSDTE outperforms two traditional churn prediction strategies, as well as three existing transfer learning strategies. Moreover, FSDTE is able to deal with the issue of imbalance class distribution in customer churn prediction better than other strategies.

Insufficiency of target domain data exists in numerous areas in CRM, such as credit scoring and fraud detection, as well as in other fields, such as intrusion detection, medical diagnostics, and information retrieval. Thus, the research results can provide important reference for other classification issues in the presence of insufficiency of customer data in target domain.

Although the proposed feature-selection-based dynamic transfer ensemble model has been successfully implemented and tested well in this study, some work can be improved further. First, in terms of time complexity, FSDTE model is a little time-consuming. To evaluate the time complexities of different models, we compared the average time consumed by ten independent runs of each model under the same conditions. The results showed that the computation time of FSDTE model on "churn" and "China-churn" datasets was 355 and 479 s, respectively, and the ranks of the time complexities for 6 models from low to high were: Bagg-OT, Bagging, TrBagg, FSDTE, TrAdaBoost, and TFS. Therefore, it is an issue worthy of studying to construct a model with better classification performance and lower time complexity under the existing framework. Second, it supposes the source domain and target domain have the same feature space in FSDTE model. Such hypothesis may not be satisfied in the real customer churn prediction. In the future, we will explore how to expand the existing FSDTE model and make it applicative for the context that the source domain and target domain have different feature spaces.

Acknowledgments Thanks to the anonymous reviewers and the editor for helpful comments on earlier version of this paper. This research is partly supported by the Natural Science Foundation of China under Grant Nos. 71101100, 70731160635, and 71273036, New Teachers' Fund for Doctor Stations, Ministry of Education under Grant No. 20110181120047, Excellent Youth fund of Sichuan University under Grant No. 2013SCU04A08, China Postdoctoral Science Foundation under Grant Nos. 2011M500418, 2012T50148 and

2013M530753, Frontier and Cross-innovation Foundation of Sichuan University under Grant No. skqy201352, Soft Science Foundation of Sichuan Province under Grant No. 2013ZR0016, Humanities and Social Sciences Youth Foundation of the Ministry of Education of PR China under Grant No. 11YJC870028, and Selfdetermined Research Funds of CCNU from the Colleges' Basic Research and Operation of MOE under Grant No. CCNU13F030.

References

1. Dyché J (2001) The CRM handbook: a business guide to customer relationship management. Addison-Wesley, Reading
2. Bhattacharya CB (1998) When customers are members: customer retention in paid membership contexts. J Acad Market Sci 26(1):31–44
3. Neslin SA, Gupta S, Kamakura W, Lu JX, Mason CH (2006) Detection defection: measuring and understanding the predictive accuracy of customer churn models. J Market Res 43(2):204–211
4. Au W, Chan KCC, Yao X (2004) A novel evolutionary data mining algorithm with applications to churn prediction. IEEE T Evol Comput 7(6):532–545
5. Kisioglu P, Topcu YI (2011) Applying Bayesian belief network approach to customer churn analysis: a case study on the telecom industry of Turkey. Expert Syst Appl 38(6):7151–7157
6. Pendharkar PC (2005) A threshold-varying artificial neural network approach for classification and its application to bankruptcy prediction problem. Comput Oper Res 32(10):2561–2582
7. Wei CP, Chiu IT (2002) Turning telecommunications call details to churn prediction: a data mining approach. Expert Syst Appl 23(2):103–112
8. Zhao Y, Li B, Li X, Liu W, Ren S (2005) Customer churn prediction using improved one-class support vector machine. In: Li X, Wang S, Dong ZY (eds) ADMA 2005, LNAI 3584. Springer, Berlin, pp 300–306
9. Wang BX, Japkowicz N (2010) Boosting support vector machines for imbalanced data sets. Knowl Inf Syst 25(1):1–20
10. Yang Q, Wu X (2006) 10 challenging problems in data mining research. Int J Inf Tech Decis 5(4):597–604
11. Verbeke W, Martens D, Mues C, Baesens B (2011) Building comprehensible customer churn prediction models with advanced rule induction techniques. Expert Syst Appl 38(3):2354–2364
12. Xia G, Jin W (2008) Model of customer churn prediction on support vector machine. Syst Eng Theor Pract 28(1):71–77
13. Lemmens A, Croux C (2006) Bagging and boosting classification trees to predict churn. J Market Res 43(2):276–286
14. Glady N, Baesens B, Croux C (2009) Modeling churn using customer lifetime value. Eur J Oper Res 197(1):402–411
15. Vapnik V (1998) Statistical learning theory. Wiley, New York
16. Pan SJ, Yang Q (2010) A survey on transfer learning. IEEE T Knowl Data En 22(10):1345–1359
17. Kittler J, Hatef M, Duin RPW, Matas J (1998) On combining classifiers. IEEE T Pattern Anal 20(3):226–239
18. Amanifard N, Nariman-Zadeh N, Borji M, Khalkhali A, Habibdoust A (2008) Modelling and Pareto optimization of heat transfer and flow coefficients in microchannels using GMDH type neural networks and genetic algorithms. Energ Convers Manag 49(2):311–325
19. Ivakhnenko AG (1976) The group method of data handling in prediction problems. Soviet Autom Contr 9(6):21–30
20. Ranawana R, Palade V (2006) Multi-classifier systems: review and a roadmap for developers. Int J Hybr Intell Syst 3(1):35–61
21. Hansen LK, Salamon P (1990) Neural network ensembles. IEEE T Pattern Anal 12(10):993–1001
22. Woods K, Kegelmeyer WP, Bowyer K (1997) Combination of multiple classifiers using local accuracy estimates. IEEE T Pattern Anal 19(4):405–410
23. Kuncheva L, Whitaker C (2003) Measures of diversity in classifier ensembles and their relationship with the ensemble accuracy. Mach Learn 51(2):181–207
24. Breiman L (1996) Bagging predictors. Mach Learn 24(2):123–140
25. Ho TK (1998) The random space method for constructing decision forests. IEEE T Pattern Anal 20(8):832–844
26. Zhu X, Wu X, Yang Y (2006) Effective classification of noisy data streams with attribute-oriented dynamic classifier selection. Knowl Inf Syst 9(3):339–363
27. Ko AHR, Sabourin R, Britto AS Jr (2008) From dynamic classifier selection to dynamic ensemble selection. Pattern Recogn 41(5):1718–1731

28. Bi W, Shi Y, Lan Z (2009) Transferred feature selection. In: Proceedings of IEEE international conference on data mining workshops, pp 416–421
29. Kamishima T, Hamasaki M, Akaho S (2009) TrBagg: a simple transfer learning method and its application to personalization in collaborative tagging. In: Proceedings of ninth IEEE international conference on data mining, Miami, FL, USA, pp 219–228
30. Dai W, Yang Q, Xue GR, Yu Y (2007) Boosting for transfer learning. In: Proceedings of the 24th international conference on machine learning, pp 193–200
31. Freund Y, Schapire RE (1997) A decision-theoretic generalization of on-line learning and an application to boosting. J Comput Syst Sci 55(1):119–139
32. Mueller JA, Lemke F (2000) Self-organising data mining: an intelligent approach to extract knowledge from data. Libri
33. Abdel-Aal RE, Elhadidy MA, Shaahid SM (2008) Modeling and forecasting the mean hourly wind speed time series using GMDH-based abductive networks. Renew Energ 34(7):1686–1699
34. Puig V, Witczak M, Nejjari F, Quevedo J, Korbicz J (2007) A GMDH neural network-based approach to passive robust fault detection using a constraint satisfaction backward test. Eng Appl Artif Intell 20:886–897
35. Xiao J, He CZ, Jiang XY, Liu DH (2010) A dynamic classifier ensemble selection approach for noise data. Inform Sci 180(18):3402–3421
36. Xiao J, Xie L, He CZ, Jiang XY (2012) Dynamic classifier ensemble model for customer classification with imbalanced class distribution. Expert Syst Appl 39(3):3668–3675
37. He CZ (2005) Self-organising data mining and economic forecasting. Science Publish, Beijing
38. Merz C, Murphy P (1995) UCI repository of machine learning databases. http://www.ics.uci.edu/~mlearn/MLRepository.html
39. Friedman JH (2003) On multivariate goodness-of-fit and two-sample testing. In: Proceedings of Phystat 2003. SLAC, Stanford, CA, pp 1–3
40. Cortes C, Vapnik V (1995) Support vector networks. Mach Learn 20:273–297
41. Tsymbal A, Puuronen S, Patterson DW (2003) Ensemble feature selection with the simple Bayesian classification. Inform Fusion 4(2):87–100
42. Doumpos M, Zopounidis C (2004) A multicriteria classification approach based on pairwise comparisons. Eur J Oper Res 158(2):378–389
43. Van den Poel D, Buckinx W (2005) Predicting online-purchasing behaviour. Eur J Oper Res 166(2):557–575
44. McNemar Q (1947) Note on the sampling error of differences between correlated proportions and percentages. Psychometrica 12:153–157

Author Biographies

Jin Xiao received his Ph.D. degree from Business School, Sichuan University, Chengdu, China, in 2010. Currently, he is an assistant professor at Business School of Sichuan University and a postdoctoral research fellow at Chinese Academy of Sciences. His research interest includes business intelligence, data mining, customer relationship management, and knowledge management.

Yi Xiao received his Ph.D. degree from School of Information Management, Central China Normal University, Wuhan, China, in 2009. Currently, he is an associate professor at School of Information Management of Central China Normal University and a postdoctoral research fellow at Chinese Academy of Sciences. His research interest includes business intelligence, knowledge management, and economic forecasting.

Anqiang Huang received his MS degree from School of Management, University of Chinese Academy of Sciences, Beijing, China, in 2010. Currently, he is a Ph.D. Candidate at School of Economics and Management, Beihang University. His research interest includes knowledge management and economic forecasting.

Dunhu Liu received his Ph.D. degree from Business School, Sichuan University, Chengdu, China, in 2010. Currently, he is an associate professor at Management Faculty, Chengdu University of Information Technology. His research interest includes achievements transformation, the alliances between industry, academia, and the research community.

Shouyang Wang received the Ph.D. degree in operations research from Institute of Systems Science, Chinese Academy of Sciences (CAS), Beijing, China, in 1986. Currently, he is a Bairen Distinguished Professor of Management Science at Academy of Mathematics and Systems Sciences, CAS, and the President of International Society of Knowledge and Systems Sciences. He has published 18 books and over 210 papers in leading journals. His current research interests include financial engineering, economic forecasting, soft computing, and decision support systems. Dr. Wang is the Editor-in-Chief or a coeditor of 12 journals including*Information and Management*, and *Energy Economics*.