DME Mini Project Report

- Marketing: Predicting Customer Churn

Hao Zhang s0916563,

Pengfei Gao s1144374,

Qiming Zhang s1153864

Abstract

The goal of this project is to compare a number of machine learning methods on a real data set. Our data set was used in the KDD Cup 2009 which is used to predict customer behaviour, such as churn (whether they will switch providers), appetency (whether they will buy new products), and up-selling (whether they may be willing to buy upgrades). As the types of the labels for those user behaviours are all the same, we mainly focus on the up-selling. Having reviewed several papers published in the proceedings of KDD-Cup 2009 competition, we found 5 classifiers which are proved to be efficient in these three tasks. We perform the five classifiers on all three tasks and compare the performances.

1. Overview of the task

This task is to predict the probabilities of customers to switch providers (churn), buy new products or services (appetency), or by upgrades or add-ons (up-selling), from the customer data provided by the Orange Company.

The data set is made up by 100,000 instances and separated into two equally sized training and test sets. There are two data set, one is large which contains 15,000 features which contain 260 categorical features, and the other contains 230 features which contains 40 categorical features. This data is collected from the customers and all features were scrambled to protect customers' privacy.

Performance is judged based on the average Area Under the ROC Curve (AUC) score of three tasks. The best result of the small or large data set will be considered as the final score.

In this report, we describe our various successful and failed attempts mainly on the small data set. Since we cannot use test data, we use cross validation on train set for measuring performance.

2. Previous work

There are 1299 teams participated in the 2009 KDD Cup and 7865 valid entries were submitted by 453 different teams [2]. The following part describes the methods used by the top 5 teams who won the prizes.

The overall winner is IBM Research which used an ensemble of a wide variety of classifiers such as boosted decision trees, random forest, and logistic regression. They mainly focused on the preprocessing such as encode the categorical features, replacing missing values and extra features constructed. However, another team, David Slate & Peter Frey, just using a simple preprocessing including discretizing and grouping of modalities and filter feature selection, followed with ensembles of decision trees, also achieved very good results.

University of Melbourne used a cross-validation method for classification, boosting with decision trees and shrinkage using a Bernoulli loss. Their performance on the small data set was very good and only less than the overall winner, IBM Research.

ID Analytics, Inc. is the only team using a wrapper feature selection following a filter. Their classifier is based on commercial TreeNet software using an additive boosting decision tree technology. They also used Bagging to gain additional robustness.

The best result achieved by IBM Research is shown in the following table.

Table 1 Result achieved by IBM Research

Churn	Appetency	Up-selling	Average
0.7651	0.8853	0.9092	0.8521

[2] analysed the valid results submitted by the participants and they find the churn task seems to be the most difficult one as the performance is much lower than the others. The up-selling task is the easiest one as the median performance gets close to the best performance. The appetency task is intermediate difficult as its median result is above the basic method.

Several teams use cross-validation method to estimate the performance by setting 90% of the whole data set as training data and the rest for testing.

The following is the conclusion by [2] for this challenge in different stages.

Preprocessing:

Most participants replaced missing values by the mean or median or a fixed value. Some added additional features indicating the presence of missing values which helps linear classifiers to calculate the missing value by choosing a suitable weight automatically. The second popular method used for preprocessing is discretization as the data set is non-normal of the distribution of features and the extreme values exists. The simple binning (code the categorical features into 10 bins by the winner) is also very efficient and helps to avoid overfitting. PCA is rarely used and reported not helpful.

Feature selection:

The most famous methods for feature selection are feature ranking. Some use other feature selection methods such as cross-validation classification.

Classification algorithm:

The most popular method used is ensembles of decision trees. As there are lots of examples, various features types, and large amount of missing values. The second popular method is linear classifiers and especially logistic regression.

Model selection:

The winners all stated that using the on-line performance feedback on 10% of the test set for model selection is not reliable because of the variance of the data. Cross-validation (i.e. 5-fold) is widely accepted to perform model selection.

3. Exploratory Data Analysis

3.1. Preprocessing and feature construction

-Data:

Small Training data set: 50,000 instances including 230 input variables, and the target value;

Large Training data set: 50,000 instances including 15,000 input variables, and the target value.

-Hardware:

We all use laptops with 4 GB memory and Intel Core i5 processors.

-Software:

We use R and Python to preprocess the training data and then use Weka to do the training.

-Work Flow:

For small training data, missing value replacement is done is R and supervised discretization and feature selection are done in Weka. For large training data, all preprocess is done is R.

We find that there are 18 empty features in the small data set and 110 empty features and delete them using R.

The participants of this challenge were judged by the area under the ROC curve (AUC). The AUC is a graph of the true positive rate, vs. false positive rate. Weka's output contains the AUC. We judge the performances of these classifiers based on the AUC and false positive rate. Because the class labels of the dataset contains only 1 and -1. According to the following table, the percentage of -1 in the whole labels is more than 92.64%. So we think the positive rate for 1 is more important the positive rate for -1 as 1 is only a very small fraction.

Table 2 Class variable distribution

	Churn	Appetency	Up-selling
-1	46328	49110	46318
1	3672	890	3682

a. Replacement of the missing values

There are a lot of missing values for many features and a huge number of possible values for the categorical variables. Considering each instance of the data has different missing features and all instances in the same group may miss the same features, we decide to create a new feature which records the number of missing features for each instance. This has been proved by [1] as it

Suggested by the papers, missing categorical values can be considered just as a separate value. We re-encode the most k frequent values for categorical features and tune on k (10, 15, 20, and 25). Finally, we followed the winner [1] to encode only 10 most common values of each categorical attribute instead of all values, in order to avoid an explosion in many features from variables with a large vocabulary. For the missing numeric values, we first replace missing values by 0 but the performance is worse than the mean of the feature. So we followed the standard approach to impute missing values by the mean of the feature. Our work refers [3] which provide sophisticated and deliberate methods.

Missing values can be treated as a prediction problem, so we tried to using missForest in R to impute the missing value in the small data set; but the time and space cost of computation forced us to give up.

b. Normalisation

We tried to normalize the features using by dividing their range. But experiments on different classifiers performed showed that normalisation is not a good choice. The performances of Random Forest and Boosted Decision Tree get a little bit worse while Logistic regression will perform normalisation itself. Therefore we then choose not to normalize the data.

c. Discretization:

As we explore the data, many numeric features are badly distributed (such as containing extreme outliers, extreme sharp Gaussian). Moreover, feature selection would like to work better if data are categorical or nominal. Consequently, we think discretization would be helpful. We considered unsupervised discretization with equal with and equal frequency and conclude that they won't work for our data. Then we choose to use supervised discretization with better encoding. The chosen discretization algorithm is Fayyad & Irani's MDL method. Other methods in the R discretization package are considered but less likely for computation. The result turns out that many features of the obtained data are better distributed.

3.2. Feature selection

As we explore the data and [1, 2, 3], many features are similar even identical. Feature selection directly influences the performance of all classifiers. Therefore, we spent a lot of time on research and trials on feature selection. We consider methods - Correlation-based Feature Selection (CFS), Information Gain, Chi Squared Probe and Information Gain Ratio by ranking the features. We then perform decision tree classification to compare the different features selected by the above three methods. The following table shows the performance of classifiers Decision Tree on upselling task, Logistic and AdaBoost (and more) on the first 30 features.

Table 3 Feature selection method comparison

	Information Gain	Chi Squared	Info Gain Ratio	CFS
Decision Tree	0.711	0.697	0.732	0.69
Logistic	0.794	0.792	0.789	0.703
AdaBoostM1	0.843	0.843	0.843	0.823

Information Gain Ratio outperforms the other three methods as its average score is the highest. According to more experiments, Information Gain Ratio is slightly better than Information Gain. CFS performs less better, which is different from our expectation.

We then select first 30, 75, 100, 150 features generated by Info Gain Ratio and the whole set of features for testing. The performances of these five feature sets using the same classifier AdaBoost on upselling task are almost the same.

Table 4 Feature number comparison

	30 features	75 features	100 features	212 features
ROC Area	0.843	0.833	0.843	0.688
True Positive Rate	0.99	0.974	0.99	0.991
True Negative Rate	0.436	0.485	0.436	0.398

We find that both first 30 and 100 features can achieve a best AUC, but 75 features can reach a best true negative rate. Considering the percentage of the value 1s of upselling class variable is less than 7.36% which means there are very little samples about the false value comparing to the true value, we think the true negative rate is very important to judge the performance because a high true negative rate indicate our classifier can actually predict the negative situations (target value = 1). So we decide to use 75 features for the classification.

4 Experiments and Results

In this section, we present the work process. There are two stages. In the first stage, we focus on finding best classifiers for upselling task and building ensemble classifiers with them. On the second stage, we focus on applying the techniques to the other two tasks - churn and appetency. All the experiments are done on the same dataset which contains 75 features. The validation is based on 4-folds cross validation.

Stage 1

According to the literature review and background knowledge, we agreed to adopt the following Weka functions to train on the dataset.

1. AdaBoost is the class for boosting a classifier using Freund & Schapire's Adaboost M1 method.
2. Bagging is the bagging result of ADTree.
3. Liblinear is the classifier for wrapped Liblinear tools. We adopt the L2 regularized logistic regression. The data is normalized while the estimation of probability is generated instead of -1/+1.
4. Logistic regression is the classifier for building and using a multinomial logistic regression model with a ridge estimator.
5. Random forest is the classifier for constructing a forest of random trees. We optimized the parameters by setting the number of features to be 30 while the number of trees to be 20.

We firstly train the dataset with the up-selling label which is the simplest according to [2]. The result is listed in table 5. We can see that the logistic regression performs the best while bagging of ADTree outperforms all the others. The reason why logistic regression is optimal is that the majority of the data are numerical. It outperforms L2 regularized logistic regression probably because the features are already selected according to the weights of influence to the class variable and regularization will result in a biased classifier. The bagging achieved a second best result probably because bagging provides additional robustness to Decision Tree classifiers.

Table 5 Best result of every classifier for up-selling

	AdaBoost	Bagging	Liblinear	Logistic regression	Random Forest
ROC Area	0.833	0.859	0.826	0.865	0.814
True Positive Rate	0.974	0.991	0.992	0.991	0.991
True Negative Rate	0.515	0.443	0.723	0.56	0.576

We use vote classifier as the meta ensemble method. As we have a limited set of classifiers, we compare the performances of three vote ensemble with three classifiers sets in table 6.

Table 6 Ensemble results

	Vote of above five classifiers	Vote of bagging and logistic regression	Vote for AdaBoost, Liblinear and Random Forest
ROC Area	0.855	0.865	0.843
True Positive Rate	0.992	0.992	0.992
True Negative Rate	0.57	0.436	0.428

The ensembles do not obtain the best performance. Logistic regression classifier alone is still the best classifier. However, the vote ensemble of AdaBoost, Liblinear and Random Forest outperforms any individual classifier of these three. Assume we can have more classifiers and more sets of classifiers, we may get a best ensemble whose bias and variance is low.

Stage 2

For tasks - churn and appetency, we use the supervised discretization and Information Gain Ratio to obtain a well processed selected dataset respectively, which contains 75 features. We exploited the optimized parameters for classifiers from the first stage; then use the same set of classifiers for tasks - churn and appetency.

The result for churn is listed in table 7. We can see Logistic Regression achieved the best performance followed by Bagging.

Table 7 Best result of every classifier for churn

	AdaBoost	Bagging	Liblinear	Logistic regression	Random Forest
ROC Area	0.71	0.72	0.678	0.724	0.655
True Positive Rate	0.927	0.999	1	0.999	0.992
True Negative Rate	0.397	0.006	0.0001	0.009	0.044

Notice that for each classifier, the true negative rate is too low while the true positive rate is nearly 1 which means it is almost the same with guessing all values positive. This is due the fact that the dataset is highly imbalanced. To balance this imbalanced dataset, we adopt CostSensitive meta classifier which introduces cost-sensitivity by reweighting training instances according to the total cost assigned to each class. We can tune on the cost matrix parameters in order to achieve larger True Negative Rate and better AUC. We tested on four sets of cost matrix parameters. However, the result has been slightly changed. Functions such as bagging are getting

worse however the result for logistic regression is improved for 0.001. The example is shown in table 8.

Table 8 Cost sensitive output for bagging and Logistic regression

	Bagging	Logistic regression
ROC Area	0.717	0.725
True Positive Rate	0.958	0.993
True Negative Rate	0.192	0.046

From the knowledge learned about, we test the classifier balanced by cost sensitive at appetency. The result is presented in table 9.

Table 9 Best result of every classifier for appetency

	AdaBoost	Bagging	Liblinear	Logistic regression	Random Forest
ROC Area	0.806	0.817	0.815	0.817	0.707
True Positive Rate	0.956	0.964	0.971	0.971	0.999
True Negative Rate	0.297	0.283	0.244	0.254	0.006

We managed to handle the large data as well. However, due to hardware limits, we face shut-down problem when processing data in R. Therefore, we use the first chunk of training dataset to do feature selection. Known the selected features names/indexes, we applied to the whole training dataset. Consequently, we succeed in processing once by applying the same procedure as we did in small dataset. To reduce the dimension of the data, we only select top 75 informative features. The results are listed in Table 10.

Table 10 Best result of every classifier for up-selling in large dataset

	AdaBoost	Bagging	Liblinear	Logistic regression	Random Forest
ROC Area	0.831	0.847	0.822	0.863	0.817
True Positive Rate	0.969	0.991	0.991	0.992	0.991
True Negative Rate	0.508	0.423	0.719	0.551	0.563

5 Conclusions

In this project, we tried to solve a real-world data mining problem - the 2009 KDD Cup Orange Challenge. In the project we went through the whole problem procedures of data mining: data explore, data preprocessing, mode selection and evaluation. We first preprocess the data and use information gain to select the features. Then we compare mainly Adaboost Decision Tree, Logistic regression, Random Forest, Bagging and Liblinear as single classifiers, and Ensemble Selection combine the five classifiers. The first three classifiers are proved to be efficient by the winner of the challenge. In this task, these three classifiers actually perform better performances than others. Based on the data we processed, the best classifier for upselling is the logistic regression followed by bagging of decision trees; the best classifier for churn is logistic regression followed by bagging of decision trees and Adaboost; the best classifier for appetency is the bagging of decision trees and logistic regression. Overall, we think data preprocessing especially feature selection plays the most significant factor in achieving the optimal performance.

References:

[1] IBM Research. *Winning the KDD cup orange challenge with ensemble selection*. In *JMLR W&CP*, volume 7, KDD cup 2009, Paris, 2009.

[2] Isabelle Guyon, Vincent Lemaire, Marc Boullé, Gideon Dror and David Vogel. *Analysis of the KDD Cup 2009: Fast Scoring on a Large Orange Customer Database*. In *JMLR W&CP*, volume 7, KDD cup 2009, Paris, 2009.

[3] Hugh Miller, Sandy Clarke, Stephen Lane, Andrew Lonie, David Lazaridis, Slave Petrovski and Owen Jones. Predicting customer behaviour: *The University of Melbourne's KDD Cup report*. In *JMLR W&CP*, volume 7, KDD cup 2009, Paris, 2009.