CS 412 Project Report

Shuanglong Wang **swang157**
Qi Wang **qiwang12**
Jatin Gupta **jgupta4**

December 6, 2014

Best Score Achieved

204	↑31	sublimotion	0.23062	5	Tue, 15 Nov 2011 21:05:42	
205	↑53	Forbin	0.23020	2	Thu, 13 Oct 2011 23:54:27	
206	↓12	Romka	0.23017	26	Sat, 31 Dec 2011 15:30:44 (-72.8d)	
207	↓11	Jagruti	0.23010	3	Thu, 05 Jan 2012 20:07:08 (-4h)	
-		**Shuanglong Wang**	**0.23007**	-	**Sun, 07 Dec 2014 00:34:56**	Post-Deadline
Post-Deadline Entry If you would have submitted this entry during the competition, you would have been around here on the leaderboard.						
208	↓20	Storm	0.23000	13	Wed, 04 Jan 2012 17:54:46 (-2.9d)	
209	↑8	max.power	0.22999	20	Wed, 04 Jan 2012 23:20:58 (-9.2d)	
210	↑12	georgewu	0.22987	9	Mon, 02 Jan 2012 06:41:09 (-29.7h)	
211	↑17	Black Jack	0.22980	17	Tue, 27 Dec 2011 12:23:17 (-7d)	

Data Preprocessing

1. Use R to change the 'NULL' and blank valuse in:
    - SubModel
    - Color
    - Transmission
    - Nationality
    - Size
    - TopThreeAmericanname
    - WheelType
    - PREIMEUNIT

- AUCGUART

to '-1'

2. Use R to fill in the 'NULL' and 0 values in:
   - MMRAcquisitionAuctionAveragePrice
   - MMRAcquisitionAuctionCleanPrice
   - MMRAcquisitionRetailAveragePrice
   - MMRAcquisitonRetailCleanPrice
   - MMRCurrentAuctionAveragePrice
   - MMRCurrentAuctionCleanPrice
   - MMRCurrentRetailAveragePrice
   - MMRCurrentRetailCleanPrice

   to the corresponding average value in each column.
3. delete column 'PurchaseDate'(hard to deal with) and 'WheelTypeID'(redundant with 'WheelType'), store the cleaned dataset as 'train_clean.csv' and 'test_clean.csv'
4. Merge both training and testing file together (a dummy 'IsBadBuy Columne') to a csv file. Use weka to read it and store the result as a .arff file. Then the possible value for all the discrete data is acquired.
5. Based on the .arff file, create two filse: 'attr_type' and 'disc_attr_possible_values'. The first one is used to store the name of attribute for training and their type {continuous or discrete}. The second one is used to store the possible values for discrete type attributes.
6. The four file mentioned above are located in the 'data' folder, all of them are used to run our program.

Classification Algorithm and Parameter Tunning

We develop three classification algorithm: Naive Bayesian, Decision Tree, and Imbalanced Adaboosting Decision Tree. The score for submission come from the third one.

1. **Naive Bayesian**
   The Naive Bayes classifier performs probabilistic prediction and follows the standard Bayes' Theorem. It assumes each that the value of a particular feature is unrelated to the presence of any other feature. The data is split according to its type i.e. categorical or continuous-valued attributes. For categorical attributes, we apply the Bayes' rule and evaluate the probabilities of individual features that are later used with Bayes' rule. For continuous-valued attributes, we assume that the features follow a gaussian

distribution and we evaluate the probabilities accordingly. We achieve 0.149 on this algorithm.

391	↓15	Agma	0.15079	3	Wed, 30 Nov 2011 03:09:59 (-7.2h)	
392	–	bkh	0.15038	4	Mon, 02 Jan 2012 09:14:52	
393	↓4	Vellerefond	0.15038	12	Thu, 05 Jan 2012 17:47:10 (-6.9d)	
-		**Shuanglong Wang**	**0.14946**	-	**Sun, 07 Dec 2014 01:44:37**	Post-Deadline
Post-Deadline Entry If you would have submitted this entry during the competition, you would have been around here on the leaderboard.						
394	↑1	Henrik	0.14868	3	Thu, 06 Oct 2011 03:34:45	
395	↑31	haveaniceday	0.14689	1	Tue, 25 Oct 2011 22:11:47	
396	↑1	Mikhail Mokhin	0.14622	3	Tue, 03 Jan 2012 22:53:23	
397	↓4	akfd	0.14535	9	Thu, 03 Nov 2011 15:14:13 (-7.2d)	
398	↓10	oooooo	0.14525	10	Fri, 07 Oct 2011 01:41:00 (-3.8d)	

2. **Decision Tree**
The decision tree follows the structure introduced in text book. At each node, we select an attribute maximize some gain. For our program we choose Information Ratio as a criteria:

$$GainRatio(A) = \frac{Gain(A)}{SplitInfo_A(D)} = \frac{Info(D) - Info_A(D)}{-\sum_{j=1}^{v} \frac{|D_j|}{|D|} \times log_2(\frac{|D_j|}{|D|})}$$

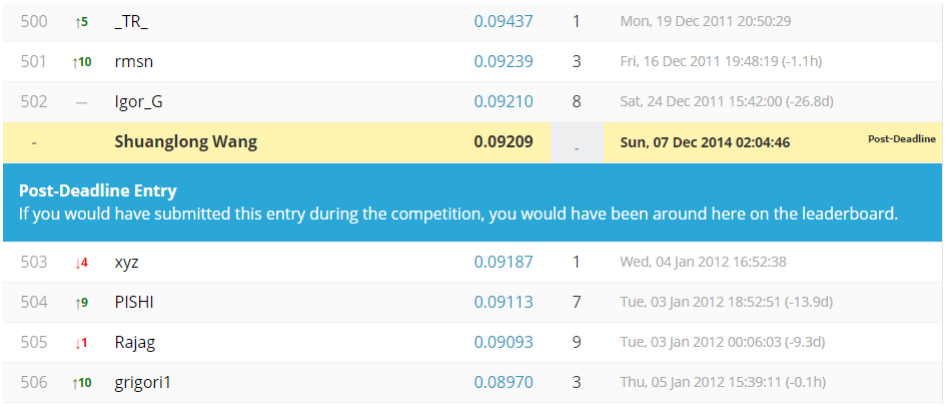
For discrete attributes, the split is done within their possible values. And for continuous attributes, a binary split point is assigned. The data set is first split according to their labels, and the conditional mean and variance of the continuous attribute is calculated on both datasets. Then the split point is given as:

$$SplitPoint_A = \frac{std(A|0) * mean(A|1) + std(A|1) * mean(A|0)}{std(A|0) + std(A|1)}$$

The rational here is that we assume that the conditional distribution on the label is approximately normal and this rough split is a MLE classification which may give a better information gain.

The stopping criteria for this algorithm is the maximum depth. After observing the result given by Weka with pruning and out empirical test, a max-depth of 3 would be suitable for this project.

The score for one single tree is about 0.09 which is pretty low. But we know that the tree is an unbiased classifier with high variance, then adaboosting with more sampling scheme will improve its performance.



500	↑5	_TR_	0.09437	1	Mon, 19 Dec 2011 20:50:29	
501	↑10	rmsn	0.09239	3	Fri, 16 Dec 2011 19:48:19 (-1.1h)	
502	–	Igor_G	0.09210	8	Sat, 24 Dec 2011 15:42:00 (-26.8d)	
-		**Shuanglong Wang**	**0.09209**	-	**Sun, 07 Dec 2014 02:04:46**	Post-Deadline

Post-Deadline Entry
If you would have submitted this entry during the competition, you would have been around here on the leaderboard.

503	↓4	xyz	0.09187	1	Wed, 04 Jan 2012 16:52:38
504	↑9	PISHI	0.09113	7	Tue, 03 Jan 2012 18:52:51 (-13.9d)
505	↓1	Rajag	0.09093	9	Tue, 03 Jan 2012 00:06:03 (-9.3d)
506	↑10	grigori1	0.08970	3	Thu, 05 Jan 2012 15:39:11 (-0.1h)

3. **Imbalabced Adaboosting Decision Tree**
The basic framework for our ensemble method is adaboosting. But because we are facing a quite imbalanced dataset. We used a modification of AdaC2.M1 (adaboosting incorporating cost-sensitive learning). Given the decision tree classifierh_t, for each iteration, the error rate is calculated as:

$$error_t = \frac{\sum_{i:h_t \neq y_i} c_i}{\sum_i c_i}$$

where c_i is an imbalance coefficient assigned to each class.

Then the weight for each data tuple is updated and normalized by:

$$W^{t+1}(i) = W^t(i) \frac{error_t}{1 - error_t} c_i \quad for\ h_t(i) \neq y_i$$

We can assign bigger wait for '1' data to let the tree do more training on that. And after some experiment, we choose 1.4 for c_1 and 1.0 for c_0. Typically, this adaboosting algorithm would converge in no more than 5 iterations in terms of error rate. Because the weight updating scheme here is exponential, kind of too aggressive, and will let the data stick to a fixed part of data. The score we received here is typically from 0.16 to 0.18.

373	–	Moritz	0.16258	23	Tue, 20 Dec 2011 10:18:45 (-9.7h)	
374	–	lapakshi	0.16134	9	Mon, 17 Oct 2011 18:08:02 (-0.5h)	
375	↑15	Matthew Roos	0.16063	2	Fri, 16 Dec 2011 04:16:30	
-		**Shuanglong Wang**	**0.16031**	.	**Sun, 07 Dec 2014 01:43:28**	Post-Deadline

Post-Deadline Entry
If you would have submitted this entry during the competition, you would have been around here on the leaderboard.

376	↓7	YaoPau	0.15941	9	Sat, 17 Dec 2011 06:05:41	
377	↑9	TPD98125	0.15891	17	Thu, 05 Jan 2012 08:39:29 (-15.9d)	
378	↑5	TPD98122	0.15888	28	Thu, 05 Jan 2012 08:06:35 (-15.9d)	
379	–	TPD66232	0.15832	1	Thu, 05 Jan 2012 15:32:30	

Finally, since we are assessed on Gini coefficient where the ranking of probability score is crucial. But our ababoosying algorithm are good only up to a few iterations. The last refinement we made is to ran this algorithm for 30 times and take the average of their probability prediction. The score get beyond 0.23.

Submission Result

I run my Adaboosting algorithm with imbalance coefficent(c_1) varying between (1,2) for 30 times, take the average of their probability prediction for each test data tuple as follows:

204	↑31	sublimotion	0.23062	5	Tue, 15 Nov 2011 21:05:42	
205	↑53	Forbin	0.23020	2	Thu, 13 Oct 2011 23:54:27	
206	↓12	Romka	0.23017	26	Sat, 31 Dec 2011 15:30:44 (-72.8d)	
207	↓11	Jagruti	0.23010	3	Thu, 05 Jan 2012 20:07:08 (-4h)	
-		**Shuanglong Wang**	**0.23007**	.	**Sun, 07 Dec 2014 00:34:56**	Post-Deadline

Post-Deadline Entry
If you would have submitted this entry during the competition, you would have been around here on the leaderboard.

208	↓20	Storm	0.23000	13	Wed, 04 Jan 2012 17:54:46 (-2.9d)	
209	↑8	max.power	0.22999	20	Wed, 04 Jan 2012 23:20:58 (-9.2d)	
210	↑12	georgewu	0.22987	9	Mon, 02 Jan 2012 06:41:09 (-29.7h)	
211	↑17	Black Jack	0.22980	17	Tue, 27 Dec 2011 12:23:17 (-7d)	

Analysis for improvement

1. The splitting method for continuous is limited to binary and the splitting point is roughly estimated. We do not know how would this influence the result, especially when we are comparing the gain ratio across continuous and discrete attribute equally.

It could be better if we found a way to search a good splitting method for continuous attributes.

2. The running time for one adaboosting decision tree is about half an hour. We generate all our results in parallel for more than 10 hours and take the average. This is definitely not a good way in reality. We hope we could find a way to improve the efficiency of our algorithm.

Work Distribution

1. Shuanglong Wang: Data preprocessing, Imbalanced adaboosting Implementation, Parameter tunning
2. Qi Wang: Decision Tree Implementation, Data processing
3. Jatin Gupta: Naive Bayes Implementation, Weka pre-testing