A novel approach based on ensemble learning for fraud detection in mobile advertising

Kasun S. Perera BPERERA@MASDAR.AC.AE
Bijay Neupane BNEUPANE@MASDAR.AC.AE
Mustafa Amir Faisal MFAISAL@MASDAR.AC.AE
Zeyar Aung ZAUNG@MASDAR.AC.AE
Wei Lee Woon WWOON@MASDAR.AC.AE
Masdar Institute of Science and Technology
Abu Dhabi, United Arab Emirates

Abstract

By diverting funds away from legitimate partners, click fraud represents a serious drain on advertising budgets and can seriously harm the viability of the internet advertising market. As such, fraud detection algorithms which can identify fraudulent behavior based on user click patterns are extremely valuable. In this paper we propose a novel approach for click fraud detection which is based on a set of new features derived from existing attributes. The proposed model is evaluated in terms of the resulting precision, recall and the area Under the ROC curve. A final method based on 6 different learning algorithms proved to be stable with respect to all 3 performance indicators. Our final method shows improved results on training, validation and test dataset, thus demonstrating its generalizability to different datasets.

1. Introduction

Smart phones are becoming increasingly popular amongst people of all ages around the world, with new applications and devices being introduced each year. In addition, successive generations of devices possess ever improving internet browsing capabilities, which in turn has resulted in profound changes in the internet usage patterns of large numbers of people, with an increasing proportion of users preferring to access the internet using mobile devices rather than desktop or laptop computers. Similarly, advertising companies around the world have shifted their focus from conventional PCs to mobile computing platforms such as smart phones and tablets. However, in line with this shift, fraudulent activities that were once perpetuated using conventional PCs have now also started appearing amongst the ranks of mobile internet users.

The pay-per-click model that was employed in the conventional internet surfing model has been transferred to the mobile sphere. In the pay-per-click model, publishers are paid by the amount of internet traffic that they are able to drive to a company's web site. This in turn is measured by the number of "clicks" received by banners and links associated with their partner accounts. This model gives an incentive for dishonest publishers to generate clicks on advertisements on their websites using manual and/or automated techniques. Dishonest advertisers might also generate false clicks on their competitor's advertisements to drain their advertising budgets Metwally et al. (2005). These clicks may be generated either manually or through the use of software that is installed on a mobile phone or a PC.

Though advertising-related fraud has increased greatly over time, perpetuators of fraud still represent only a tiny fraction of the large community of online advertising driven publishers. Unfortunately, the activities of this small proportion of fraudulent publishers can inflict a very high cost on the profitability of advertisers and other service providers (for example search engines). Thus many research efforts have been undertaken and many approaches have been examined to model the behavior of and subsequently to identify these fraudulent publishers. In spite of this, no single model can detect these fraudulent publishers with 100% accuracy. In particular, fraud detection in the case of mobile advertising is even harder since many of the features and parameters found on conventional computer systems are not available on mobile phone networks.

1.1. Problem formulation

Due to the limitations of mobile phone networks, new methods are needed to evaluate the fraudulent behavior of ad publishers. New features have to be created using existing parameters to capture the behavior of the publishers. Previous studies Kantardzic et al. (2008); Li et al. (2011) show that these fraud publishers try to act rationally when simulating clicking behavior. However, despite (or perhaps due to) these endeavors, their click pattern still deviate from the typical click patterns of legitimate publishers.

In this work we used two data sets provided by Buzzcity to analyze the behavior of publishers and classify/identify fraudulent publishers from legitimate publishers. First data set contains the publisher information as publisher ID, account no, address and status (in training set only). The second data set contains click details of all above publishers. These details include click id, publisher id, ip, agent, category, country, campaign id and referrer. Since the provided raw data cannot be used directly with any models for classification with sufficient accuracy, we use methods describe in the following sections to add meanings to data set, aggregate data sets over publisher information and define the behavior of each publisher. These formatted data then can be used for classification with methods currently available.

2. Data Preprocessing and Feature Creation

2.1. Datasets

The experiment was performed using three sets of data. The first two sets were used for training and validation. The training set contains two files, *Clicktrain* and *Publishertrain*, where *Clicktrain* includes 8 different attributes and has 3,173,834 instances. Each instance represents a click record for a partner. *Publishertrain* contains the records for each of the 3,081 legitimate partners along with their labels (fraudulent or legitimate). The validation set also contains similar data with 2,689,005 instances in the *Clickvalidation* file, while the *Publishervalidation* file contains records for 3,064 partners (but without the labels). The prediction model was built, trained and validated using these two data sets. The final model was tested on the test data set. The test data set also contains two files: *clicktest*, which contains 2,598,815 instances and *publishertest* with 2,112 instances.

2.2. Preprocessing

In this study precise and careful data pre-processing and feature creation was a critical step which greatly improved the overall accuracy of the model. As a first step, the given data was visualized, which included attributes like iplong, agent, partnered, campaign ID, country, time-at, category, referrer in "click" dataset and partnered, Address, bank account and label in "partner" dataset. Each attribute in the data was analyzed and evaluated in terms of its effect towards modeling behavior of a partner. What seemed clear was that not all of the features would be useful; as such, certain attributes like partner address, bank account from feature creation process were excluded from the modeling process.

After selecting the attributes which were to be considered for further consideration, the *trainingpartner* data set was merged with the *traningclick* data set to have the status of each partner in click dataset. Each partner can have one of three possible statuses: "OK" for legitimate partners, "Fraud" for fraudulent partners and "observation", which indicates that the final status of the partner is still unknown and that he/she is still being evaluated. To better deal with this, the data was converted into three modified datasets. In the first dataset all partners labeled as Observation were re-labeled as "OK". In the second dataset, all partners labeled as "Observation" were re-labeled as "Fraud". For the third dataset all three labels were retained. Training and testing were performed accordingly on all three datasets.

2.3. Feature Extraction

Feature extraction is another pre-processing step which can strongly affect the accuracy of the model. Features are numerical indices that quantify the behavior and patterns of every partner. Thus, a properly selected feature should be able to capture properties or trends which are specific to fraudulent partners and be robust towards the evolving patterns of behavior adopted by the said partners.

In order to create features from the selected attributes, we refer to the literature and identify some features that have been used in the Google adsense fraud detection mechanism. Though Google does not reveal the exact features used, the literature provides basic features that can be used in any fraud detection algorithm. We also referred to other published research on fraud detection in conventional systems as fraud detection in mobile advertisements is still emerging.

To create features, we select each attribute separately and try to model the partner's click pattern with respect to that attribute by creating many parameters based on that particular attribute. We identify these parameters as maximum, average, skewness, variance etc. we try to create as many features as we can, thus allowing us to capture many characteristics of a partner. In the feature creation phase we were not concerned about the dimensionality of the dataset as a feature selection procedure would be applied later in the process. Details of the feature creation methods applied to different attributes are as follows.

Attribute: Time-At

Fraudulent partners will often disguise their activities using a variety of tricks such as generating very sparse click sequences, changes in IP addresses, issuing clicks from different computers in different countries and so on. Others stick to the traditional approach of only generating the maximum number of clicks in a given interval. It is important that any prediction system is able to recognize both these attempts at concealment. A number of additional features were derived from the time-at attribute from the given dataset, with the number of clicks for each partner over different pre-defined time intervals such as 1 min, 5 min, 1 hours, 3hours and 6 hours. The intention was to capture both the short and long term behaviors of the partners. These features were selected as partners often try to act rationally and have constant clicks in very sparse time intervals. We also record the number of clicks each partner receives over different time intervals and then aggregate these values using different measures. We calculated Maximum clicks, Average click, Skewness and Variance in click pattern of each partner for a given time interval. Click Variance for each time interval provide information about click pattern of the partner whereas Skewness helps to determine the deviation of number of clicks from the average clicks of the partner in a defined time interval. The following figure1 shows all the features we derive from one single attribute 'time-at' from given original dataset.

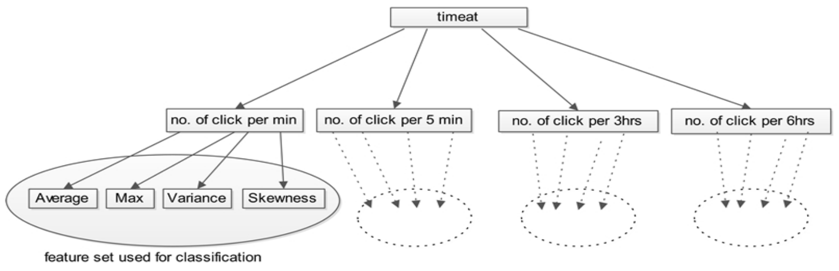

Figure 1: Feature creation from time-at attribute.

Attribute: iplong

IP address is another attribute that can be used to characterize the behavior of a partner, since it is a reflection of the number of computers/mobile devices used or different times at which the user clicks on a particular advertisement. Since many IP addresses are dynamically allocated when users connects via an ISP, it is not unusual for the same user to have different IP addresses. For the given time period of 3 days we observed changes in IP addresses and number of clicks from a given ip address for a given partner id. Then we use parametric measures over IP address attribute (iplong) to define the behavior of a partner. The following features were created using the iplong attribute.

1. MaxSameIPClicks = for a given partner, count the number of clicks from all different IP addresses he/she have, then obtain the maximum of that
2. NoOfIPs = for a given partner, count the number of clicks from all unique IP addresses he/she have.
3. Click/IP ratio = total clicks for a given partner / total unique ip address associated with that partner.

4. EntropySameIPClicks = for a given partner, count the number of clicks from all different IP addresses he/she have, then obtain the Entropy of that.

5. VarSameIPClicks = for a given partner, count the number of clicks from all different IP addresses he/she have, then obtain the Variance of that.

Some fraudulent partners may try to increase their reward by clicking repeatedly on an advertisement but all of these clicks might come from the same ip. Many clicks originating from the same ip or an unusually large click to ip ratio can be a sign of fraudulent behavior and might place the associated partner under suspicion. Lower variance in the number of clicks from each ip is indicative of a legitimate partner whereas higher variances might indicate a fraudulent partner. Similarly the entropy for the distribution of the number of clicks originating from each IP can be another useful indicator of fraudulent activity.

Attribute: Agent

The Agent attribute is the phone model that user used to browse the web and eventually make clicks on advertisements. As mentioned above, a particular fraudulent user might use one phone, but with many dynamically allocated IP addresses. Thus, we use following measures to derive features from 'agent' attribute in the set of attributes. The features created using agent attribute are MaxSameAgentClicks, MaxSameAgentClicks, VarSameAgentClicks, SkewnessSameAgentClicks. These features also calculated using similar method as defined in iplong attribute.

Similarly we define features on Country, Campaign ID as well. When we analyze category with partners, we found that each partner is assigned to only one category, thus we avoid taking category to derive more attributes. But instead we define the prior probability of being fraud for particular category based on the training dataset. We obtained number of users assigned for a given category and then found the number of fraudulent partners in that set to obtain the prior probability of being fraud for a given category. For referrer attribute we derived, referrer/click ratio by obtaining referred clicks over total number of clicks for a given partner. At the end of feature creation process we had 41 different features created from different individual and set of attributes from the data set. The list of those 41 features is provided in Appendix A (available at: `www.dnagroup.org/PDF/FDMA12_TeamMasdar_AppendixA.pdf`). These carefully crafted features always have threat of over fitting the model, which can be solved by feature selection method.

2.4. Feature Selection

Feature selection is another important step towards building a robust prediction and classification model, and can help to prevent overfitting of the data. We tested our model with different feature selection techniques including Principal Component Analysis (PCA), Common Spatial Patterns (CSP), and wrapper subset evaluation. Of the approaches tested the wrapper method Kohavi and John (1997) resulted in the highest accuracy compared to the other methods.

The wrapper method implements a greedy search algorithm for finding the subset of features in space of all feature, where each subset is evaluated in terms of the resulting accuracy when used with the classification model or learning algorithm (in each case, 10-fold cross validation is used to obtain a reliable estimate of accuracy). We also trained and tested our model without performing any feature selection. The results and observations from both approaches will be discuss in later sections.

3. Experimental Configuration

Our experiments were conducted using the Matlab (`www.mathworks.com/products/matlab`) numerical computing environment on a computer with 8GB memory and 12 processors. As the raw dataset contains 3 million rows of data, it took a long time to load the dataset into the Matlab environment. Thus, different approaches were explored to optimize our experimental configuration. We found that SQL is a better choice for creating features from attributes, thus we used MySQL (`www.mysql.com`) server to load the data and queried every feature we needed from the database. Through a single script file run on MYSQL server, we were able to generate a CSV file that contained the partners together with 41 features which defined their behavior. This CSV file was then used in WEKA (`www.cs.waikato.ac.nz/ml/weka`) to train, validate and test different models to find the best model. We selected WEKA as it is very user friendly and supports many learning and classification algorithms, and also because of the limited time the we had for our experiments.

3.1. Methods Used

Due to the importance of fraud detection a variety of solutions have been proposed, but none of these solutions work perfectly and produce sufficient accuracy under all conditions. Some solutions work well on one dataset and badly on others. Thus, new methods for fraud detection are always being proposed and tested.

Fraud detection is another type of classification which has its own special characteristics. There were many approaches proposed for fraud detection which claim to have higher accuracy. Our approach is to use traditional classification models over data derived from click data. We can tune the parameters in these models to suit with the behavior of our data. It was important that any model used could generalized to work with training, validation and test data sets and subsequently with any other data set containing the same features. In order to achieve a stable model, we tried a few different models and over a range of different model parameters; we also selected subsets of features and chose the algorithm with the highest precision, the highest ROC and with a high degree of consistency.

A number of different methods were tried including decision trees, regression trees, artificial neural networks and support vector machines. For each method we also used different learning algorithms, thus each evaluation model is in fact a unique combination of a given classification technique and learning algorithm. After analyzing the results on training and validation data, we found that the decision tree technique was particularly promising and provided very good accuracy. As such, subsequent discussions will cover only decision tree based models.

As mentioned earlier, in any given situation, only a relatively small fraction of the partners are going to be guilty of fraud. As such, any fraud detection data set will inevitably be highly skewed, where the number of instances in the "OK" class will greatly outnumber the number of instances in the "Fraud" and "Observation" classes. In our dataset the percentage of Fraud, Observation and OK was 2.336, 2.596 and 95.068 respectively. The skewed nature of the data forced the prediction model to be biased towards the class with higher population. To deal with the skewed data we used different methods such as resampling and SMOTE. Sampling usually maintains the population of one class and increases (via resampling of the minority class) or decreases (by sub-sampling the majority class) the population of the other class depending on the type of sampling used. In our experiments we tried both sampling methods followed by randomization of instances. Results obtained both with and without the use of sampling methods are discussed in the results section.

Bagging and boosting are promising ensemble learners that improve the results of any decision tree based learning algorithm. In our research we used Bagging for all the decision tree leaning algorithms. Bagging aggregates multiple hypotheses by the same learning algorithm invoked over different distributions of training data Breiman (1996). Bagging helps to generate classifiers with smaller errors on training data as it combines the different hypotheses which individually have a large prediction errors. Metacost which is a form of bagging with cost associated with each training instance to minimize the expected cost was also used as a weak learner with our learning algorithms. We also used random sub space, and logiboost learners with decision tree models.

Based on the results obtained we selected Bagging with decision tree models as the best and most consistent method for classification. A decision tree is a tree like structure, where the classification process starts from a root node and is split on every subsequent step based on the features and their values. The exact structure of a given decision tree is determined by a tree induction algorithm; there are a number of different induction algorithms which are based on different splitting criteria like information gain, gain ratio and the Gini coefficient. As the tree is constructed, over-fitting may occur Sahin and Duman (2011). Thus, we performed tree pruning to evaluate the performance of the tree and avoid over fitting.

After setting up both the weak algorithms and the main learning algorithm, the resulting models were evaluated using training and validations datasets. After evaluating the results of the different models 6 different models were chosen that obtained the best overall values in Precision, Recall, and Area Under the ROC Curve (AUC). The main purpose of evaluating all three measures is to build a model which can detect a high percentage of fraud cases while maintaining a high degree of precision. The general classification model used is shown in Figure 2.

4. Results

In this section we describe the training and validation results obtained using the approaches mentioned in the previous section. Various decision tree based learning algorithms were analyzed for use as weak learning algorithms in combination with Bagging and Metacost methods. The below are main experiments we performed.

Using the algorithms listed in Table 1, different methods were implemented and evaluated to finally reach the most stable model. Those methods are described below. The average precision mentioned below is the one obtained with the validation data. The results were obtained by submitting the predicted results to the competition website.

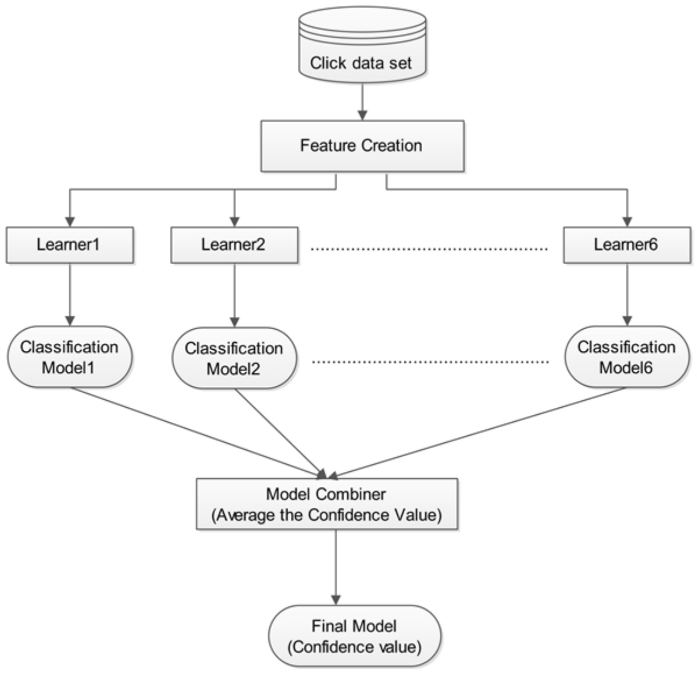

Figure 2: Classification model.

Decision Tree Learner	Ensemble Learner
J48	Bagging, Metacost, Logiboost, Random Subspace
Repetition Tree	Bagging, Metacost, Logiboost
Random Forest	Bagging, Metacost, Logiboost

Table 1: Decision tree algorithms and corresponding meta-learning algorithms.

Results with/without feature selection: As mentioned earlier we have generated many features with the training and validation sets provided. To ensure not to make over fitting of the data on the model we have evaluated different algorithm listed above with and without feature selection. We found no significant differences in the results obtained using both approach. The accuracy obtained when using feature selection was actually slightly lower than when no feature selection was used. For example, the average value of precision obtained using j48 as a learning algorithm without feature selection was 45% whereas with feature selection the score was 44.82%. This approach was attempted with all the algorithms and in each case similar results were obtained. One possible reason for this observation is that decision tree algorithms incorporate tree pruning routines which already function as a form of feature selection. As such, inclusion of an additional feature selection stage will be of limited benefit (or, as was observed in this case, could actually result in a slight drop in accuracy of the resulting classifier).

Results with/without sampling: As the data was highly skewed, some measures for balancing the data was necessary. We tried to balance the data using two common techniques which were resampling and SMOTE. Resampling and SMOTE performed very well with the training set but performed very badly with the validation set. Results without sampling or SMOTE were more than 20% better than the results obtained using sampling and SMOTE.

Results from 2-class/3-class classifications: Different models were created using datasets containing only 2 classes ("OK" and "Fraud"), and all 3 classes as ("OK", "Observation" and "Fraud"). For the 2-class setting, two approaches were taken. In the first approach, all "Observation" cases were converted to "OK", while in the second approach all cases of "Observation" were regarded as "Fraud" cases. In the 3-class setting we simply used the labels provided with the data. Of all the 3 techniques, the dataset with 2 labels where "Observation" was converted to "Ok" performed better than the other two. The precision obtained using the J48 tree as the learner and with 2 classes ("Observation" $\rightarrow$ "OK") was 50.37%, with 2 classes ("Observation" $\rightarrow$ "Fraud") was 46.1% and with all 3 classes was 45%. Thus we can see that converting all of the "Observation" cases to "OK" and without feature selection as described above was the best approach which gave the highest precision.

Results from individual algorithms: We evaluated the prediction performance of different algorithms for all data sets. Few algorithms which gave best result alone are mentioned above. There were many algorithms with very low true positive and false negative rates, and which thus were able to obtain very high precision scores. These algorithms were able to obtain high precision because of their low false positive value. We were only interested on algorithms which have high true positive rates and precision. The precisions of the different algorithms when applied on the the two class dataset is j48:50.37%, Reptree: 46.82%, and Logiboost: 44.82%.

Results from combination of different learning algorithms (final approach): With our approach we were able to pass the benchmark score but none of the algorithms alone was able to obtain precision higher than 50.37%. We analyzed the results and found that every algorithm has a drawback, which was either a high false positive rate or a lower rate of true positives. This indicated that choosing any one of the algorithms represented a trade-off between high sensitivity on the one hand, and higher precision on the other. Thus for our next step we combined the results from the different algorithms trained using the 2-class dataset and without feature selection. 6 different algorithms were chosen which obtained higher values for precision, recall and AUC when evaluated alone. This method proved to be the best as we obtained 59.39% precision with the validation set; it also performed well with the final test set, achieving a score of 0.41.

5. Discussion and Conclusion

As mentioned earlier, to conceal their illicit activities, fraudulent partners often try to act rationally so as to resemble legitimate users. We scanned through the features related to fraudulent partners to identify common patterns and to get insights into fraudulent behavior. With the features generated with time-at attribute, we observed that about 90% of the fraudulent partners ($\sim$63) have very small numbers of clicks within the 1min, 5min or 3hr intervals. Also the variance and skewness is very low, which shows that most fraudulent partners operate using small numbers of clicks for the 1 minute – 3 hour time intervals. Though this activity might help them to hide amongst legitimate partners it was observed that most of their clicks sequences were very systematic, which could thus be a sign that these partners as fraudulent. With the attribute ‘agent’, we observed very high variance on agents which indicated that many fraudulent partners tried to use large numbers of agents (mobile phone models) to act as a rational user.

Fraud detection in the pay-per-click advertisement model is extremely important as this model is open to abuse by unscrupulous users. Compared to conventional computer systems, fraud detection on mobile phone networks is harder since access to user click information is limited. In this research we perform an experiment to detect fraudulent partners based on click data associated with mobile phone internet surfing. We generate new features based on the attributes, and use these features to model the behavior of each partner. To achieve a stable prediction model, we perform various feature evaluation techniques, classification algorithms together with different ensemble learners. To generalize the prediction model we combined the results obtained using the 6 learning algorithms which performed best on the training and validation sets. Each algorithms’ performance was evaluated based on the average precision score obtained by submitting the results to the competition website. The final results showed that our model performed well with different datasets, and was able to detect a very high number of fraudulent partners.

However, due to time constraints we did not examine feature creation by merging two or more attributes over partners. As future work we are looking forward to identifying more features based on the combined attributes which best describe the behavior of a partner. More experiments on feature selection techniques over created features are needed to identify the best features and hence avoid over-fitting.

References

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

M. Kantardzic, C. Walgampaya, B. Wenerstrom, O. Lozitskiy, S. Higgins, and D. King. Improving click fraud detection by real time data fusion. In *Proc. 2008 IEEE International Symposium on Signal Processing and Information Technology (ISSPIT’08)*, pages 69–74, 2008.

R. Kohavi and G. H. John. Wrappers for feature subset selection. *Artificial Intelligence*, 97:273–324, 1997.

X. Li, Y. Liu, and D. Zeng. Publisher click fraud in the pay-per-click advertising market: Incentives and consequences. In *Proc. 2011 IEEE International Conference on Intelligence and Security Informatics (ISI’11)*, pages 207–209, 2011.

A. Metwally, D. Agrawal, and A. El Abbadi. Duplicate detection in click streams. In *Proc. 14th ACM International Conference on world Wide Web (WWW’05)*, pages 12–21, 2005.

Y. Sahin and E. Duman. Detecting credit card fraud by decision trees and support vector machines. In *Proc. 2011 International MultiConference of Engineers and Computer Scientists (IMECS’11) Vol. I*, pages 1–6, 2011.