A Class Imbalance Learning Approach to Fraud Detection in Online Advertising

By

Baruhupolage Kasun Perera

A Thesis Presented to the

Masdar Institute of Science and Technology

in Partial Fulfillment of the Requirements for the Degree of

Master of Science

in

Computing and Information Science

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. These fraudulent behaviors exist in credit card transactions, insurance claims, business transactions and new fields such as smart grids. Fraud detection systems suffer an inherent problem of Imbalance Class Distribution which need to be addressed as conventional classification algorithms fails on this scenario. In this thesis we propose a novel approach for click fraud detection which is based on a combination of different models with Ensemble Learning as well as Sampling Methods. The proposed model is evaluated in terms of the resulting precision, recall and the F-Measure. A final method based on 4 different learning algorithms proved to be stable with respect to all performance indicators. Our final method shows improved results on training, validation and test datasets, thus demonstrating its generalizability to different datasets.

This research was supported by the Government of Abu Dhabi to help fulfill the vision of the late President Sheikh Zayed Bin Sultan Al Nayhan for sustainable development and empowerment of the UAE and humankind.

Acknowledgments

At the very beginning I would like to thank my parents and my family for supporting me to reach my ambitions. Without their support I would not be able to have a peaceful mind which in return makes me to drive on the way to success.

Despite the capability of a Ship and its Captain, a proper navigation system is necessary to reach its destination on the silent sea. Dr. Zeyar Aung who is the founding pillar in my success and the advisor of my tenure at Masdar Institute, gave an enormous support to success in my research as well as other educational matters. His guidance, patience, novel ideas helped me to gain deep understanding about the subject.

Besides my advisor, I would like to thank Dr Wei Lee Woon for his great support in many crucial points in my research, as reviewing my work and continuous suggestions provided and for being a Research Supervisory Committee member. I also would like to thank Dr Jacob Crandall for being my Research Supervisory Committee member and support he offered me throughout the period.

Finally I would like to thank my friends in Masdar Institute who are consisting over 30 countries, for their help for me to feel like home, and support they offered at every single moment at UAE.

Baruhupolage Kasun Perera,

Masdar City, May 13, 2013.

Contents

List of Tables

List of Figures

CHAPTER 1

Introduction

With the advancement of technology and media, many markets become deregulated where customers bid for the goods and services they use. Also in order to gain new customers, the companies use many advertising methods ranging from conventional paper advertisements to new ads on social media such as Facebook or Twitter. Not all the customers are legitimate, and not all the companies are truthful. In any industry there are some fraudulent customers or any type of stakeholders which try to disrupt the system and gain some advantage. Due to the advancement in technology these fraudulent behaviors become hard to identify since they use different methods to hide themselves among the legitimate partners. Identifying these fraudulent behaviors are crucial for the smooth running of the business and to maintain the trust of legitimate stakeholders. These fraudulent behaviors exists in pay per click advertisement model as well as new fields like smart (power) grids [2, 52], where fraudulent customers tend to compromise the system by providing false data and act falsely to gain additional advantages.

Inherent features of the Pay Per Click (PPC) model make it vulnerable to fraudulent behaviors of its users and provide them incentives to gain advantages and drain money from the system. The PPC model has interesting features which make it hard to deal with fraudulent customers without compromising its important features. Thus, it is necessary to employ fraud detection mechanisms to detect fraudulent activities in the system and prevent them from gaining advantages.

A legitimate scenario of the PPC model is depicted in Figure 1.1 - Figure 1.4 where legitimate users publish advertisements on their websites. These advertisements are received from search engine or an advertiser. These advertisers or search engines make money as being a third party for real advertising companies and publishers.

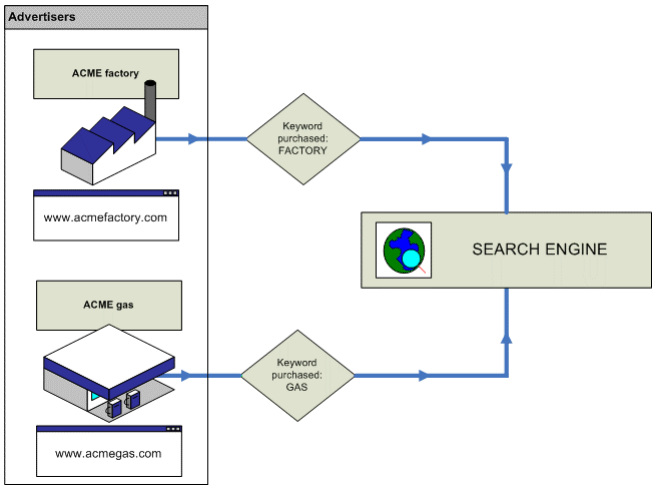


Figure 1.1: A company that wants to be advertised will purchase keywords from a search engine or provide their ads to advertisement company which then place these advertisement on publisher websites or applications [53].

The ad publisher earns money when Internet surfers visit his/her website or use application and clicks on the advertisements shown on the page. The true visitors visit these ads having interest on the products or services offered by the company. Thus after their visits, some of these surfers convert as buyers of the products or

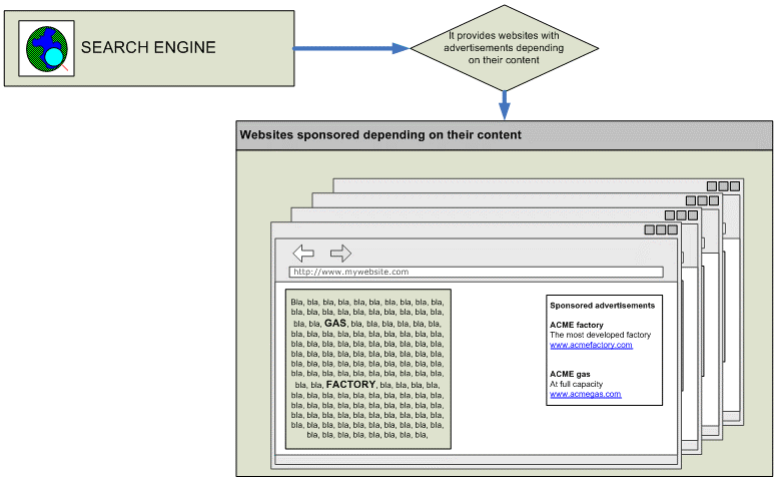


Figure 1.2: The search engine provides registered publishers with different advertisements depending on the content of webpage or application [53].

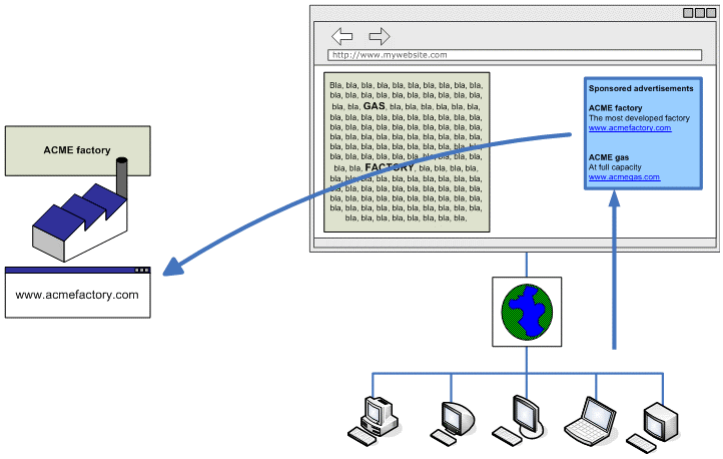


Figure 1.3: When users access those websites or applications they were shown the ads delivered to website and few interested users might click those ads with intention of buying products or services [53].

services offered. But if the publisher is fraudulent then he/she generates false clicks on the ads using ClickBots [66] or using human activities. The Figure 1.5 - Figure 1.7 depicts a scenario of fraudulent publisher activities.

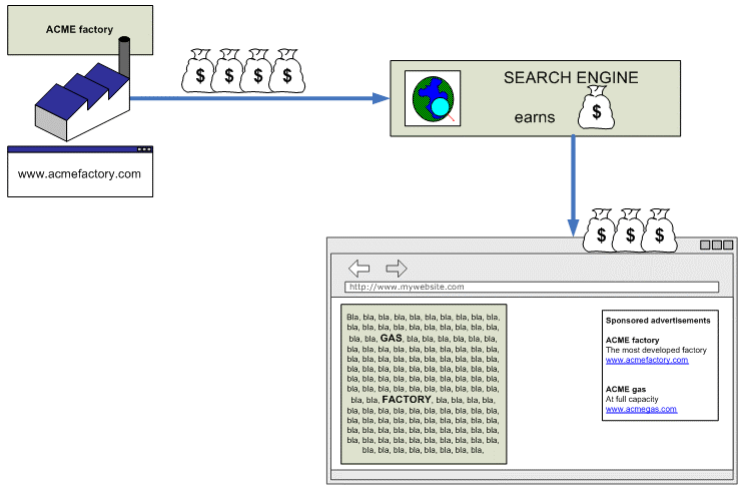


Figure 1.4: For each click made by a potential buyer, the search engine pays money to publisher. Search Engine receives this money from company for delivering ads to publishers [53].

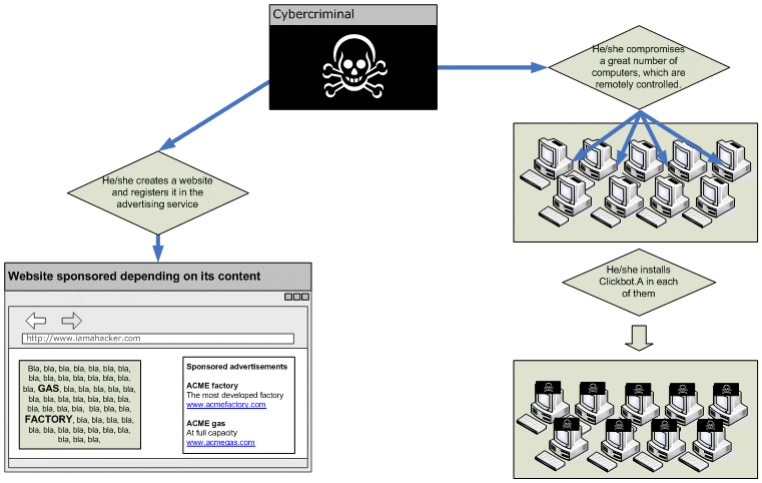


Figure 1.5: A cybercriminal registers a website or an application in the advertising company or search engine, the criminal might already compromised many computers and install clickbots [53].

1.1 Motivation

As fraudulent publishers or users exists in many fields such as online advertising or smart grid networks, identifying these fraudulent behaviors is extremely hard.

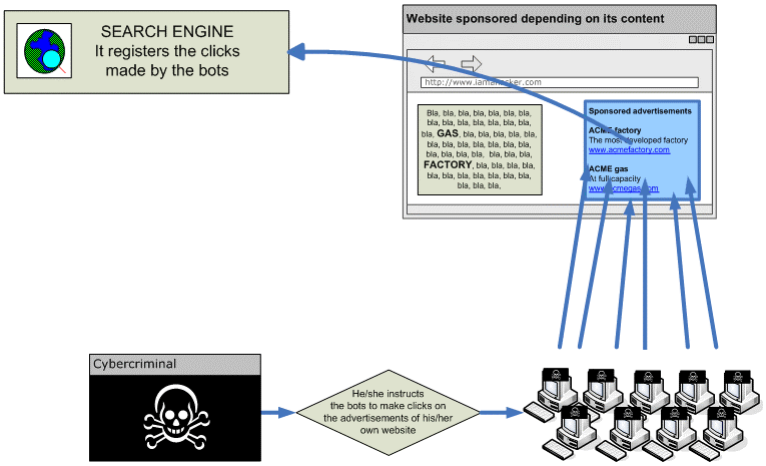


Figure 1.6: The fraudster instructs the compromised computers to make clicks on advertisements on his website or application. These clicks are carefully organized in order not to detect by fraud detection systems [53].

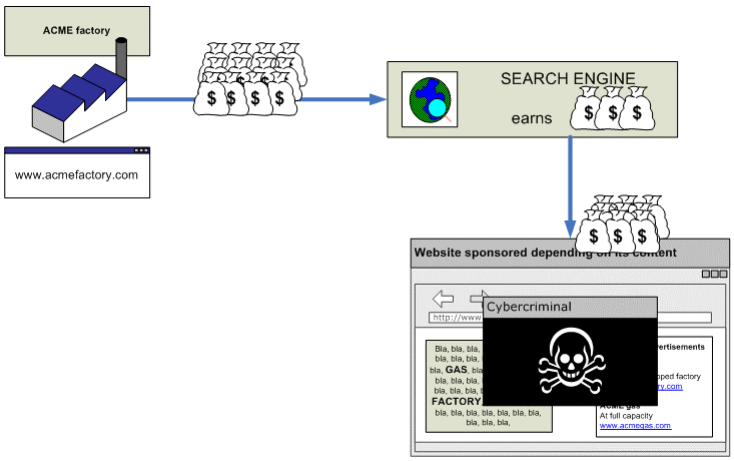


Figure 1.7: As search engine obliged to pay for the clicks it received from publisher, the criminal earns huge sum of money [53].

Even though there exists very limited number of fraudulent partners in any industry, they cost huge sums of money to the companies which offer a particular good or service. As mentioned in [29] fraudulent publishers cost billions of dollars to the advertising companies, and such activities create dissatisfaction among the legiti-

mate publishers because they cannot trust the online advertising system anymore.

As the smartphones become affordable to many people and internet service providers (ISP) provide mobile service internet packages for affordable rates, internet browsing on smartphones grows at a higher pace. Due to high performance hardware available on smartphones many companies also provide mobile games which attracts many users in many ages. Most of these games offer free versions from which companies make money by showing small advertisements in those game screens. As such mobile advertisements become increasingly popular and a crucial method of attracting customers.

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 [35, 43] show that these fraud publishers try to act rationally when simulating clicking behavior. However, despite (or perhaps due to) these endeavors, their click patterns still deviate from the typical click patterns of legitimate publishers.

Interestingly, since these fraudulent users are a small portion compared to other legitimate users, identifying these users becomes harder, as common classification algorithms tend to produce more errors when the class distribution is imbalanced. More importantly these errors are higher with minority class which is the crucial class in fraud detection. Correctly identifying a fraudulent user as a fraudulent user is more important than correctly identifying a legitimate user as a legitimate user. Also classifying a fraudulent user as a legitimate user costs more to the system than classifying a legitimate user as a fraudulent user, since any legitimate user can prove his/her authenticity. Due to the features defined above it is necessary to develop new methods to identify fraudulent behaviors when imbalanced class distribution presents in the real world.

1.2 Thesis Statement - Objectives

The objectives of this research have been identified based on the motivation and are as follows:

- Propose a data mining and machine learning based architecture to identify fraudulent customers.
- Identify behaviors or patterns of the fraudulent customers which differentiate them from legitimate customers.
- Propose possible methods to cope with the class imbalance problem

1.3 Research Contribution

In our course of research, we have analyzed click traffic data from mobile phone devices and derive a robust mechanism to classify fraudulent publishers from legitimate publishers. These methods incorporate a range of approaches tested individually and combines these methods and algorithms to achieve a highly accurate classifier.

As fraud detection suffers the inherent problem of class imbalance learning, our research provides mechanisms to deal with such situations by sampling the dataset and using an ensemble approach to correctly classify the minority class which is the important class in fraud detection or any other class imbalance learning problem.

1.4 Relevance to Masdar

Masdar Institute and Masdar as a company is interested in the renewable energy industry and these power plants soon will be a part of the smart grids that might be deploy in near future. Masdar as an energy company also provides consultancy

for the deployment of smart grids. As any industry has its own drawbacks, it is visible that smart grid networks also suffer from fraudulent access and intruder attacks [17, 40] as well as privacy leakages [41, 42]. Though these fraudulent customers will comprise a small percentage compared to all the customers, they can destroy the balance of the smart grid which then affects all other legitimate users. Identifying these fraudulent behaviors is vital, thus need to employ fraud detection mechanisms. Fraud detection has underline problem of class imbalance learning which we address in this research by combination of different approaches. Thus, this research can be extended to employ in smart grid networks where it can be used to detect the fraudulent activities on smart grids.

1.5 Thesis Organization

Rest of this thesis is organized as follows, Chapter 2 provides the background of fraud detection, detailed description of fraud detection in PPC. Latter half of Chapter 2 explains literature review on fraud detection in PPC, class imbalance learning problems and ensemble learning approaches. In Chapter 3 we describes the proposed architecture and objectives of our research. Experimental Setup explaining data set used, algorithms used, tools used are explained in Chapter 4. Next chapter, which is Chapter 5 describes our feature engineering process which is one of the most important component in our research. Chapter 6 provides Results and Analysis, followed by Chapter 7 which provides Discussion and Future Work. Chapter 8 concludes this thesis with a conclusion.

CHAPTER 2

Background and Literature Review

”Fraudsters” that can be found in many different business models refers to a abuse use of a company’s systems without leading to direct legal penalties. In current competitive business environments, fraudsters can make critical situations if they are predominant and prevention mechanisms are not fail-safe. Fraud detection which works from the back-end side is a part of the fraud control system. The fraud detection system automate the screening process of the activities of the users and use the screened data for analysis, which then decides which users are fraudulent and which users are legitimate. The importance of this area makes it a prominent system in many government and business organizations.

2.1 Background in Fraud Detection

Fraud detection research has evolved from many research groups around the world and they use different approaches for prevention and detection. The analytical sys-

tem of these proposed solutions depends on areas such as artificial intelligence, auditing databases, econometrics, fuzzy logic, machine learning and data mining, artificial immune systems, neural networks, pattern recognition, statistical computing, visualization and many others.

Fraudulent access to the system can originate from both inside and outside of the system. Inside attacks are from employees and managers of the system where as outside attacks are from external end users of the system. Figure 2.1 defines a variety of potential fraudsters in a system and their level of severity.

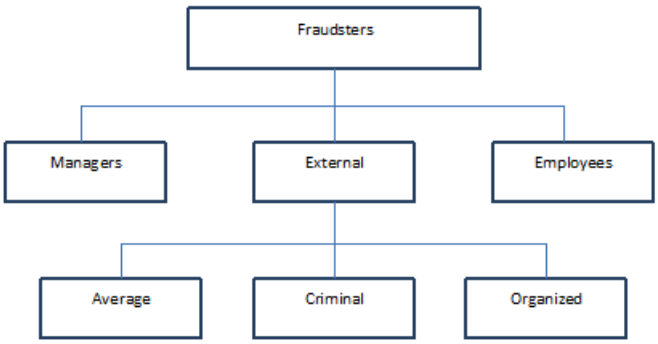


Figure 2.1: Different types of fraudsters to the system and their severity.

Any system is susceptible to inside attacks or fraudulent activities by company's employees and these are the most common fraudulent activities, which also makes it harder to prevent as these employees and managers have high level access to the system. The Fraudulent behaviors can thus divide into three categories as defined in Figure 2.1. Since managers and general employees have different access levels to the system, fraud activities from those two individuals are categorized to two separate categories. External users are can act as customers or a suppliers and commute frauds making advantages over the drawbacks and features of the system. These fraudulent activities, based on the impact can categorized to three categories as Average, Criminal and Organized.

As mentioned earlier fraudulent behaviors can originate within the organization, as such managers within the organization can produce false financial reports which need to identify. The finding on [4],[5],[21],[21],[45] explains the mechanisms to identify managerial level fraudulent behaviors within the organization. [37] provides experimental finding for fraud detection when committed by internal employees, these fraudulent activities includes abnormal retail transaction which harm the company's profits. Insurance companies are another kind of businesses which susceptible for fraudulent attacks. These fraudulent activities exists in motor insurance [1],[3],[9],[62], crop insurance [47], home insurance [6], and medical & life insurances [28],[27],[48],[70]. No matter which category it falls, detecting these fraudsters are vital for smooth run of the insurance companies. In these companies the fraudster are humans and provide false data to gain advantages.

Credit cards are highly affected by fraud activities which illegal users make false credit cards, or use credit card with fraudulent applications. [69] provide solutions for screening and detection fraudulent behaviors in credit card applications where as [12],[14],[19],[23],[38] provides screening of logs of credit card transactions and detect frauds from the dataset.

2.2 Click Fraud Detection

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 [61]. 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 [49]. 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.

2.3 Literature Review

As the Internet grows faster and access to the internet becomes widely available in many parts of the world, advertising on internet become more favorable due to its inherent features. These features includes, it's cheaper to advertise on internet with compared to conventional advertising techniques, it uses pay per click model where the company has to incur money only when someone visits the ad, with the help of internet an ad can reach to vast amount of users, and most importantly an ad can delivered to a person who has an interest towards the good or services offered by the advertising company. Due to its performance, simplicity and accountability, the PPC pricing model has become the leader in online advertising having nearly $16 billion revenue in 2010 [57].

2.3.1 Click Fraud Detection in Pay Per Click Model

Due to its inherent features, the pay per click model, poses a threat to internet advertising and it is considered as a cyber crime. This occurs when a person, automated script (autobot) or computer program mimics a legitimate user of a internet service (eg. web browser)clicking on a ad, having intention to generate a charge per click without having actual interest towards the advertising link. The high impact of these actions on online advertizing industry, and the amount of money it can drain from the companies caught the attention of academic and industry researchers to identify such threats. Many researchers have introduce several methods and techniques to identify, report, and prevent those actions. According to click forensics, the click fraud rate reached its highest level of 22.3% in 2010 [44].

Agrawal et al, presented fraud detection in traffic streams of web advertising network based on the hit inflation techniques [50]. In their research they focus on detecting publisher frauds and ignore the advertiser frauds which are caused by hit shaving [59]. Advertiser who does not pay commission for some of the traf-

fic it received from a legitimate publisher is called hit shaving. They classify the attacks to two main categories as non-coalition and coalition attacks. Non coalition attacks stands for fraudulent attacks from a single publisher, where coalition attacks are stand for more coordinated attacks by few publishers. Classification of these attacks is based on statistical analysis of traffic streams. They analyze the cookies together with IP, a click received from and use different features such as single cookie single IP, single cookie-mulitple IP, and cookie less traffics to identify the attacks. After analysis they found out that suspicious publisher websites has moderate level traffic but comes from around the globe or particularly from different IP addresses from different regions. With their system they were able to find fraudulent traffic which sums to 5.6% of the total traffic generated by all publishers.

The fraudulent clicks generated by click-bots usually have a pattern of many duplicate clicks. The clicks that have same IP address, same agent, same cookie details or if the click details are identical in a short time period is knows and duplicate clicks. In [72], a methods to identify fraud publishers based on duplicated clicks has been proposed. The proposed method employs jumping window and sliding window models to detect the duplicate clicks. They propose two main algorithms that screen the click traffic in single pass to detect the click frauds. Proposed GBF algorithm uses bloom filters works on jumping windows with small number of sub-windows. TBF algorithm proposed to use sliding windows as well as jumping windows with large number of sub-windows and the algorithm utilizes timing bloom filter so such purposes. Authors use false positive rate to evaluate the algorithm and it shows 0.007 false positive rate when used with GBF algorithm and 0.001 false positive rate when used with TBF algorithm.

Kantardzic et al, proposed real time click fraud detection system [35] which uses information from different sources, not just analyzing user click data traf-

fic. They merged service side information with client side information using multi source data fusion. In their system, they use attributes such as IP, Referrer, Java Script details, cookie details, Tracking ID and time zone to analyze the fraudulent behaviors. In their analysis, it showed that referrer is an interesting feature to detect frauds. Referrer as server side feature does not helps much to detect the frauds but when it fusion with client side data, the results improved substantially. Their system can detect the traffic from Software systems which usually stands for fraudulent click data.

An interesting paper "Cracking the smart clickbot" [66] presented by Walgampaya and Kantardzic to detect the software generated clicks. As mentioned in the introduction section, many fraudulent partners use simple software known as clickbots to automate the clicks and generate traffic to advertisers without actually seeing the page. In the paper they presented the underline architecture of clickbot and its generated click patterns. Authors also presented machine learning algorithm to detect the click patterns similar to clickbot traffic. A Bayesian classifier is used to screen the server logs and detect such click patterns eventually yield to identify fraudulent clicks. For the evaluation of the proposed method, they use Precision, Recall and F-Measure parameters.

In [13], Chertov et al. proposed a method to use Non-Dyadic wavelet to identify some click frauds, where they convert the click to a wavelet by analyzing different features associated with the click. A time domain analysis is used which they divide the 24hr day into three main periods namely working time, free time, sleep time. Then these periods subdivided in to two periods each with four hours making total periods to be six. Then pollock method was used for the experiment as it it enables dynamic dilation factor change by specifying the required sample size divisors.

2.3.2 Class Imbalance Learning

Class imbalance problems have a growing concern in recent years as it imposes classification difficulties for common classification algorithms. Thus new algorithms and techniques to deal with data are necessary to mitigate the classification errors. In class imbalance learning problems, how to recognize minority class is the key focus because it is more important and expensive than the majority class. For example in cancer cell identification, identifying a cancer cell correctly is more important than identifying a healthy cell correctly. This section provides detailed literature review on class imbalance learning as our research on fraud detection also has the same characteristics. In our dataset only $\tilde{2}.5\%$ instances are fraudulent instances and identifying this fraudulent instances is the key focus.

Due to higher inaccuracy when using common classification algorithms on class imbalanced data, researchers have introduced ensemble learning approaches to deal with such data. Though ensemble learning method is applicable to most classification algorithms, it is easy to combine with resampling techniques which is necessary in class imbalance situation. Also the combination of multiple classifiers results in a reduction of the error bias/variance as discussed in [63]

Algorithm Level Approaches

In general introducing a bias or cost sensitive methods is a one approach to deal with class imbalance problems in algorithm level. In decision tree algorithms, one approach is to adjust the probabilistic estimate at the tree leaf [58, 71]. Another approach is to develop new pruning techniques.

For SVM based methods, techniques such as using different penalty constants for different classes [46] and adjusting the class boundary based on kernel- alignment ideas are proposed. To develop an algorithmic solution, it is necessary to have the knowledge of both the corresponding classifier learning algorithm and the

application domain. Especially a thorough comprehension on why the learning algorithm fails when the class distribution of available data is uneven. This may be due to the features associate with to describe the behavior of the instances.

Some proposed a one class learning approach which it trains the algorithm to identify only a single class and others as outliers or invalid data. This approach does not try to partition the hypothesis space with boundaries that separate positive and negative examples, but it attempts to make boundaries which surround the target concept. In this approach it measure the similarity between target class and the query object and if it is within the threshold value it classify as the target class otherwise not. Both SVM and Neural Network are used as one class based techniques to classify target class. Under certain conditions such as multi-modal domains, the one-class approach is reported to be superior to discriminative (two-class learning) approaches [30]. A too strict threshold means that positive data will be sifted, while a too loose threshold will include considerable negative samples. Hence, to set up an effective threshold is crucial with this approach.

Data Level Approaches

Many solutions proposed in data level to mitigate the effects on class imbalance in classifications. These solutions includes altering the data by resampling to manage the imbalance on the dataset [16],[73],[10]. It includes randomly oversampling the minority class as well as undersampling the prevailing class. Due to the randomness of this approach, it does not produce good results in many cases. Whereas informative sampling has introduced to oversample the minority class by means of different criteria, also informative undersampling on majority class introduced by targeting the sample data to eliminate from the dataset. Researchers also focused on generating synthetic data by oversampling the minority class, and also combination of above techniques.

Though resampling is the often used method in data level when dealing with class imbalance problems, the problem lies in how to decide the correct class distribution for a given dataset. According to [68] the general conclusion is that, when class distribution is equal (1:1) it performs well with respect to the classification performance in the small class. But this distribution is not optimal in all cases, thus optimal class distribution is vary from data to data.

Optimal class distribution is not the only major problem with relate to data level approaches for class imbalance problem. When resampling is used, how efficiently and effectively resample the data is another issue. Random oversampling or undersampling are not working in many cases because random oversampling may duplicate the some parts of the data over the other part, having distort the characteristics of the original data. A more effective resampling process should be, first, detecting the sub-concepts constituting the class and then, oversampling each concept, respectively. This helps to maintain the original data distribution. This method imposes high cost on data analysis since it is necessary to have sufficient information on data to determine the sub concepts in the distribution. Thus informative oversampling is efficient but costly compared to random sampling.

Informative undersampling attend to even the distribution by selecting sub samples from majority class data. but problem lies when deciding the criteria to select the sub samples. For example when using distance based method, selecting instances that are far from the decision boundary or from the minority class may introduce more errors compared to selecting samples close to the minority class. Because these close by instances may crucial when deciding the decision boundary rather than the far instances. thus selecting far apart instances or close by instances is depends on case to case and no systematic mechanism exists to be used as general method.

Cost Based Learning

In this approach it introduces varying costs for different misclassification types. It introduces penalty of classifying samples from one class as another. For example in [51] using SVMlight [33], it introduces cost function to train the model. Let $C(i, j)$ denotes cost of predicting an instance from class i as class j. As we familiar with class imbalance situation, higher cost imposes on predicting $C(+,-)$ where classifying a positive (minority class) as negative where as lesser cost associate with $C(-,+)$ when classifying a negative instance as a positive instance. Thus, cost of misclassifying a positive instance outweigh the cost of misclassifying negative example as positive, $C(+,-) > C(-,+)$ and classifying an instance correctly has the penalty 0. When using this method, in the training process it try to minimize the number of high cost errors while maintaining total misclassification cost to be lower. A cost-sensitive classification technique takes the cost matrix into consideration during model building and generate a model that has the lowest cost [73],[20],[34] and [64].

2.3.3 Ensemble Learning

Ensemble learning is an approach of using single or many algorithms for prediction and aggregate predictions using voting methods to achieve a higher accurate classifier. Many different ensemble approaches have proposed such as Bagging, Boosting etc. Ensemble methods are heavily used when data set is imbalance as these methods provide accurate predictions as opposed to single algorithm methods.

Bagging

In bagging method, it generates multiple versions of the same predictor and aggregates the numerical prediction of these versions using plurality vote to make the

prediction class [26]. Multiple versions of the same algorithm generates by bootstrap replicated on the training data set and using these new sets as new learning sets. When the prediction is the probabilities $P(j|x)$ which specify the probability of instance x belongs to class j a different method can be used for aggregation. In such situations simple averaging over all bootstrap replications can be used and then use the estimated class arg max_j P(j—x).

Boosting

Boosting is a commonly used technique to improve the accuracy and performance of any classification algorithm. Boosting is used to reduce the error in weak leaning algorithms, and it believed to reduce the error significantly. Similar to bagging methods, boosting also works by applying a given weak algorithm repeatedly on various weighted distributions of the given training dataset. Then it combines the predictions of each of these classifiers to single composite classifier. As mentioned in [24] boosting performs better than bagging in the cases where weak learning algorithm generates simple classifiers. But when C4.5 is used as a weak learning algorithm, boosting provides slightly lower results than begging method. It also suggests that boosting can be used with simple rules to generate classifiers that are quite good relative. [36] also argues that the C4.5 algorithm itself can be treated as a boosting algorithm.

Summery, boosting works on full training dataset provided, but by changing the weights for some data which are incorrectly classified with the previous model used. In contrast bagging works on resampling the data set into N different sampling with replacement and apply same model on the different samples. The results are then combined with plurality voting.

CHAPTER 3

Proposed Architecture

3.1 Objectives

Fraud Detection in Smart Grid Network is crucial to run the electricity grid smoothly. Fraud Detection in online advertising system shows similar behavior to the fraud detection in smart grid. Since there is no data available for Smart Grid, we used data received from Buzz City which specify click data on advertisements on Buzz City Network. Since this data received from mobile phone devices, certain features of similar data on computer system networks are not available on this data.

The objective of this experiment is to identify suitable algorithms and methods to classify publishers as legitimate and fraudulent publishers. Since the dataset is very imbalanced having small number of fraudulent publishers compared to large number of legitimate publishers, this research focus on methods that can used in class imbalance learning. Since the same behavior exists in any fraud detection system where only few fraudulent users exists, similar approach can be used in

other cases too. In the Smart Grid Network, the central control system server stores the data sent by every user on the grid and this data can then be used to identify the fraudulent users. It is same as the advertising company server stores the click data for each partner (publisher) and used these data to identify fraudulent publishers. Thus this experiment will shed the lights on analyzing click data and identifying fraudulent publishers using machine learning methods.

3.2 System Architecture

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, recall, the highest F-Measure and with a high degree of consistency.

In our experiment, the proposed architecture is depicted in Figure 3.1, where we start with two databases provided and finally produce the predicted model. This model then can be used to classify any incoming fraudulent publishers using the same set of features.

As shown in Figure 3.1, our proposed fraud detection system follows a sequential architecture where it uses the click database to extract the click data and combined with publisher data in the training process. This combined data then used to generate the features that describe the behavior of each publisher. The features are generated based on the click data and analyzing click patterns. Each attribute available in the combined dataset is used to generate few features.

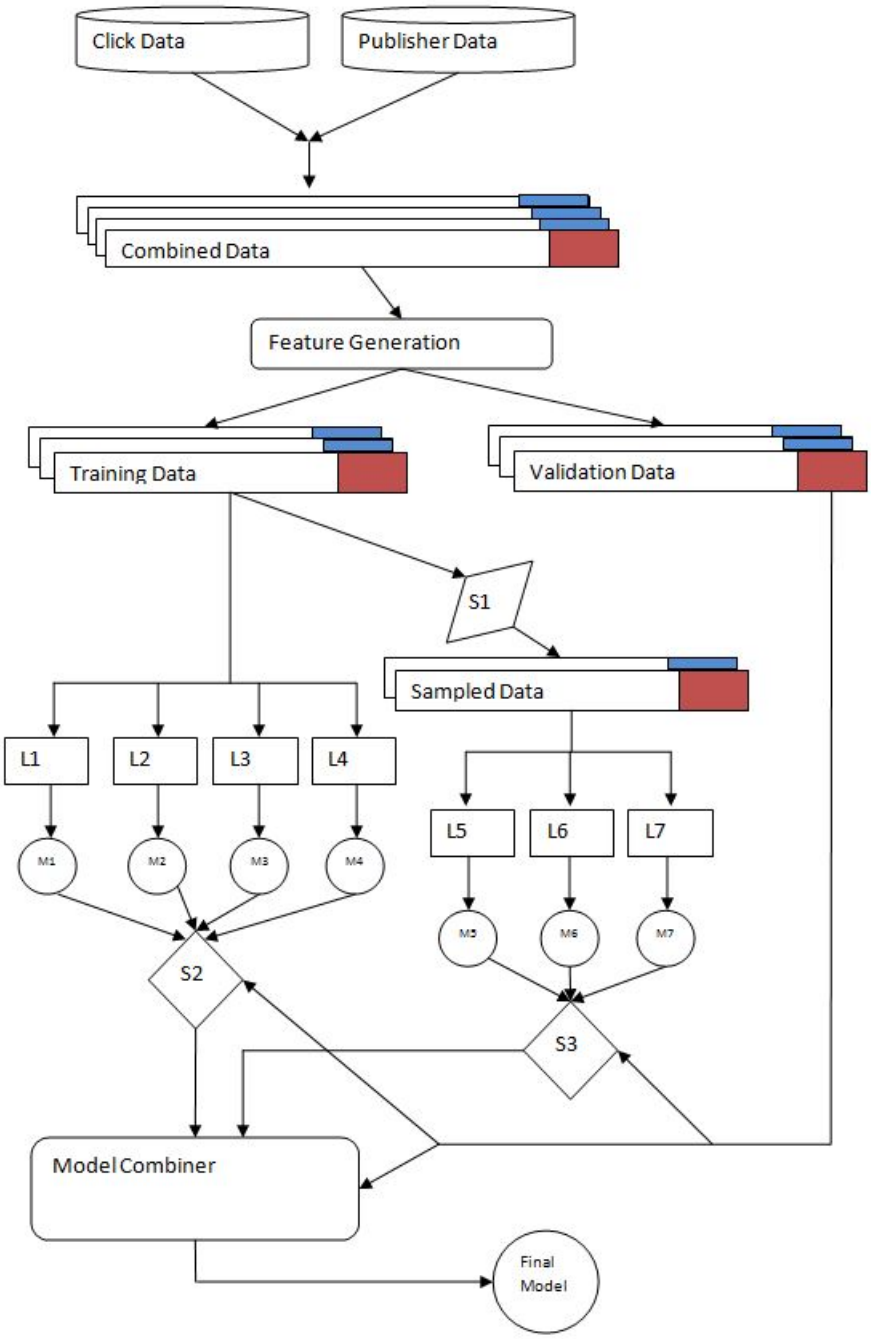


Figure 3.1: Overall architecture of the fraud detection system.

These generated features are then used to train the model. From the training dataset it extracted samples based on the distribution of the class. In our experiment, the initial training dataset contains 2.5% fraudulent partners. The selection of sample sizes is explained in the subsequent chapters. The sampling techniques such as random sampling, SMOTE [11] oversampling, SMOTE under sampling and informative undersampling can be used in this phase.

The derived model then used in the commissioner server to detect the frauds. The model screens the server log which store click data with respect to each publisher and generate reports on regular basis. The deployment of model is depicted in Figure 3.2.

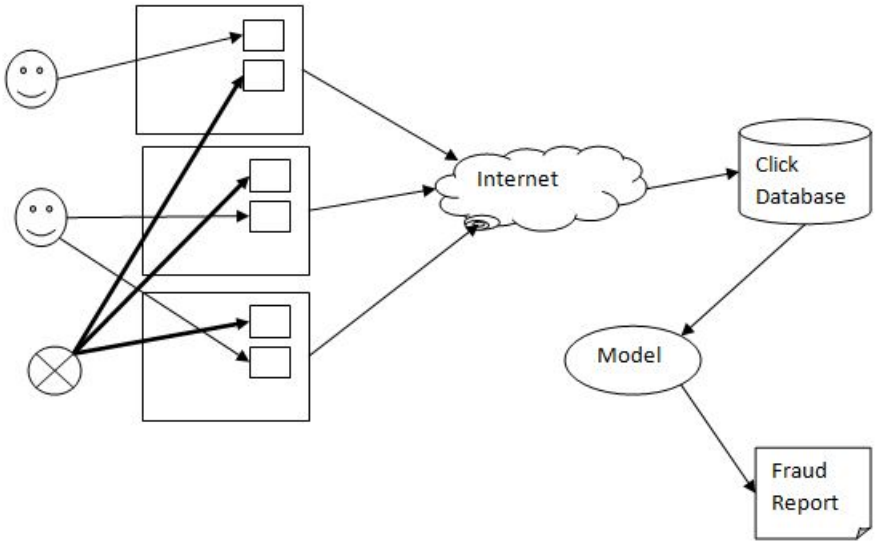


Figure 3.2: Deployment of fraud detection model at the server side of the commissioner.

The model deployed in the server side can update with new known data, which is verified after detecting from the system. Thus, the model deployed is maintained to grabs the most recent abnormalities and predict the incoming traffic based on the past known data.

CHAPTER 4

Experimental Setup

After careful analysis of data set and the objectives, we perform the experiment using publicly available tools as well as self-implemented methods. The description about data set, tools used and other related matters explained in this chapter.

4.1 Data Set

The data used in this experiment is received from Buzz City pvt ltd. BuzzCity is a global mobile advertising network that has millions of consumers around the world on mobile phones and devices. In Q1 2012, over 45 billion ad banners were delivered across the BuzzCity network consisting of more than 10,000 publisher sites which reach an average of over 300 million unique users per month. The number of Smartphones active on the network has also grown significantly. Smartphones now account for more than 32% phones that are served advertisements across the BuzzCity network.

The "raw" data used in this competition has two types: publisher database and click database, both provided in CSV format. The publisher database records the publisher's (aka partner's) profile and comprises several fields as stated in Table 4.1

Field	Description
`partnerid`	Unique identifier of a publisher
`bankaccount`	Bank account associated with a publisher (field may be empty)
`address`	Mailing address of a publisher (encrypted; field may be empty)
`status`	Label of a publisher, which falls into three categories: • `OK`: Publishers whom BuzzCity deems as having healthy traffic (or those who slipped their detection mechanisms) • `Observation`: Publishers who may have just started their traffic or their traffic statistics deviates from system wide average. BuzzCity does not have any conclusive stand with these publishers yet • `Fraud`: Publishers who are deemed as fraudulent with clear proof. BuzzCity suspends their accounts and their earnings will not be paid

Table 4.1: Fields in the publisher database.

On the other hand, the click database captures the click traffics associated with various publishers/partners. These data is the primary concern to model the behavior of the partners. The details description of click traffic data is presented in Table 4.2

The raw data provided is encrypted for privacy protection, and the format of some fields (e.g., partner id, bankaccount, campaign id, referer URL) may change over time due to different encryption schemes. Thus, our predictions made do not depend on specific field format or encryption method. The goal of this research is to build a data-driven methodology for effective detection of fraudulent publishers. In particular, the task is to detect "Fraud" publishers (positive cases) and separate them from "OK" and "Observation" publishers (negative cases), based on their click traffic and account profiles. This would help shed light on several key areas:

- What is the underlying click fraud scheme?
- What sort of concealment strategies commonly used by fraudulent parties?

Field	Description
id	Unique identifier of a particular click
iplong	Public IP address of a clicker/visitor
agent	Phone model used by a clicker/visitor
partnerid	Unique identifier of a publisher
cid	Unique identifier of a given advertisement campaign
cntr	Country from which the clicker/visitor is
timeat	Timestamp of a given click (in yyyy-mm-dd format)
referer	URL where ad banners are clicked (encrypted; field may be empty)
category	Publisher's channel type, which can be one of the following: • ad: Adult sites • co: Community • es: Entertainment and lifestyle • gd: Glamour and dating • in: Information • mc: Mobile content • mg: myGamma (BuzzCity media) • ow: myGamma offer wall (BuzzCity media) • pp: Premium portal • se: Search, portal, services

Table 4.2: Fields in the click database.

- How to interpret data for patterns of dishonest publishers and websites?
- How to build effective fraud prevention/detection plans?

Table 4.3 contains sample 3 publishers from publisher's database with their status labels. The statuses of these publishers are "Fraud" which is for fraudulent publishers, and "OK" and "Observation" for normal publishers. The status Observation is for the publishers which do not have enough traffic to classify as OK or Fraud publishers, thus consider as a "Normal" publisher.

partnerid	bankaccount	address	status
8iaxj		14vxbyt6sao00s84	Fraud
8jljr			OK

Table 4.3: Publisher sample in raw training data. There are missing values in bankaccount and address which are not interested fields in classification.

Table 4.4 lists three sample clicks from each publisher listed in Table 4.3. As shown in Table 4.4, some fields can be empty, such as referrer or agent.

id	iplong	agent	partnerid	cid	cntr	timeat	category	referer
13417867	3648406743	GT-I9100	8iaxj	8fj2j	ru	2012-02-09 00:00:00	ad	26okyx5i82hws84o
13417870	3756963656	Samsung_S5233	8jljr	8geyk	in	2012-02-09 00:00:00	es	15vynjr7rm00gw0g
13417872	693232332	SonyEricsson_K70	8jljr	8gkkx	ke	2012-02-09 00:00:00	es	
13417893	2884200452	Nokia_6300	8jljr	8gp95	vn	2012-02-09 00:00:01	es	
13418096	3648406743	GT-I9100	8iaxj	8fj2m	ru	2012-02-09 00:00:08	ad	24w9x4d25ts00400
13418395	781347853	GT-I9003	8iaxj	8fj2j	ru	2012-02-09 00:00:20	ad	4im401arl30gc0gk

Table 4.4: Click sample in raw training data. There are missing values in `referer` and `agent`. The raw features include encoded IP address of a clicker (`iplong`), mobile device model used by the visitor (`agent`), campaign ID of a particular advertisement campaign (`cid`), country of the visitor (`cntr`), type of publisher (`category`), or an URL where the ad banner is clicked (`referer`). Other raw features such as browser type, operating system, and carrier information are not provided.

Table 4.5 summarize the count statistics of the publishers and clicks. As shown, the publishers and clicks data were each split into three sets: train set (for building the predictive model), validation set (for model selection and optimization), and test set(for evaluating models' generalization). Each dataset comes from a different time window, so there could be publishers (publisherid) who appear across different sets and whose status label may change. You can infer the time windows from the timestamps of the click dataset.

			No. of publishers			
Dataset	Time period	No. of clicks	Fraud	Observation	OK	Total
Train	9-11 Feb 2012	3,173,834	72	80	2,929	3,081
	(Thu to Sat)		(2.34%)	(2.60%)	(95.07%)	
Validation	23-25 Feb 2012	2,689,005	85	84	2,895	3,064
	(Thu to Sat)		(2.77%)	(2.74%)	(94.48%)	
Test	8-10 Mar 2012	2,598,815	82	67	2,847	3,000
	(Thu to Sat)		(2.73%)	(2.23%)	(94.90%)	

Table 4.5: Statistics of the original data.

Each click dataset captures the click traffics over a 3 day period, while each publisher dataset records the publishers who received at least one click during that period. Note that the Fraud and Observation publishers constitute very small portions of the population in comparison to the OK publishers, thus rendering this problem challenging for contemporary classification techniques.

4.2 Tools Used

This section provides brief overview of tools used in our experiment and their capabilities. We have used WEKA, SVM^{light} and MATLAB in our experiment.

4.2.1 WEKA

WEKA is a tool developed by Wikato University and it is a collection of implementation of machine learning algorithms for data preprocessing, classification, clustering, regression and association rule mining. It also provides the data visualization functions and the tool is freely available to download and use for academic purposes.

4.2.2 SVM^{light}

SVM^{light} is a machine learning tool developed in C for Support Vector Machines [Vapnik, 1995]. The main features of the program includes fast optimization algorithms with,

- working set selection based on steepest feasible descent
- ”shrinking” heuristic
- caching of kernel evaluations
- use of folding in the linear case

Other useful features that available in SVM^{light} are,

- solves classification and regression problems. For multivariate and structured outputs use.
- solves ranking problems
- computes XiAlpha-estimates of the error rate, the precision, and the recall

- efficiently computes Leave-One-Out estimates of the error rate, the precision, and the recall
- includes algorithm for approximately training large transductive SVMs (TSVMs)
- can train SVMs with cost models and example dependent costs
- allows restarts from specified vector of dual variables
- handles many thousands of support vectors
- handles several hundred-thousands of training examples
- supports standard kernel functions and lets you define your own
- uses sparse vector representation

The optimization algorithms used in SVM^{light} are described in [33], [31]. The algorithm has scalable memory requirements and can handle problems with many thousands of support vectors efficiently. The software also provides methods for assessing the generalization performance efficiently. It includes two efficient estimation methods for both error rate and precision/recall. XiAlpha-estimates [33],[32] can be computed at essentially no computational expense, but they are conservatively biased. Almost unbiased estimates provide leave-one-out testing. SVMlight exploits that the results of most leave-one-outs (often more than 99%) are predetermined and need not be computed [33]. SVM^{light} can also train SVMs with cost models [51] which is also used in our experiment.

4.2.3 MATLAB

Matlab is proprietary software and a high level technical language that is available for numerical computation, visualization, and programming. It is easy to analyze

data, develop algorithms, programs to classify, cluster or visualize data in MATLAB. It also provides interfaces to other programming languages such as C/C++ and JAVA. Matlab has rich set of toolkits, which are predefined commonly used algorithms and techniques. These toolkits are easy to use and help to reduce the time and increase the quality of the work.

4.3 Evaluation Procedure

For evaluation of algorithms we used two step evaluation on the provided dataset. As the provided datset contains 3 sets of data known as Training, Validation and Test sets, we first use Training set to train models and then use the validation set to test the accuracy of the model. Based on the validation set results we update the model, retrain and re-evaluate on the validation set. Once the results cannot be improved further with validation set, we selects that model to be used in model combiner which we used to combine many models to achieve higher accuracy. Figure 4.1 provides basic architecture of the evaluation procedure.

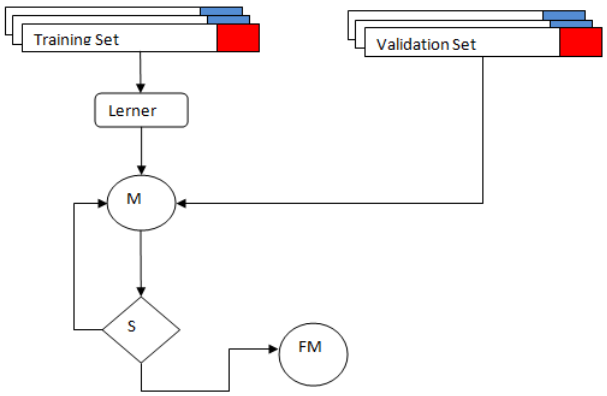


Figure 4.1: Model Evaluation Procedure.

4.4 Evaluation Metrics

It is necessary to evaluate the algorithms and methods used by certain evaluation criteria. The most commonly used criteria to evaluate the performance are Precision and Recall. But in Fraud Detection scenario where there are many legitimate publishers and very few fraudulent publishers it is hard to use just Precision and Recall measures only.

4.4.1 Precision, Recall, Accuracy and the True Negative Rate

The following table 4.6 shows the possible outcomes with the actual outcomes related to predictions. The Precision, Recall, Accuracy and True Negative Rate have defined as below using the information specified in the table.

		Actual Class (Observation)	
		Normal	Anomaly
Predicted Class (Expectation)	Normal	TP (True Positive) Correct Result	FP (False Positive) Unexpected Result
	Anomaly	FN (False Negative) Missing Result	TN (True Negative) Correct Absence of result

Table 4.6: Confusion matrix.

Precision

$$Precision = \frac{TP}{TP+FP} \tag{4.1}$$

Recall

$$Recall = \frac{TP}{TP+FN} \tag{4.2}$$

Accuracy

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \tag{4.3}$$

True Negative Rate

$$TNR = \frac{TN}{TN + FP} \tag{4.4}$$

F-Measure

F-Measure [65] also known as F1 score is used to evaluate the performance of a model in different class distribution scenarios. When the class distribution is balanced one can use F1 score and when F2 is used it weights recall two times higher than precision. Thus for imbalance class distribution problems F_β can be used where β stands for the weight of recall over precision.

$$F1 \quad Score = 2 \quad \times \quad \frac{precision \quad \times \quad recall}{precision \quad + \quad recall} \tag{4.5}$$

$$F1 \quad Score = (1+\beta^2) \quad \times \quad \frac{precision \quad \times \quad recall}{\beta^2.precision \quad + \quad recall} \tag{4.6}$$

Average Precision

For performance evaluation, we chose to use the *average precision* criterion, which favors algorithms capable of ranking the few useful items ahead of the rest. Such criterion is particularly suitable for detecting rare instances such as fraudulent publishers [74]. We briefly describe the criterion as follows: Let π be the number of relevant (i.e., actual `Fraud`) instances. Among the $t \times 100\%$ top-ranked instances, supposing $h(t) \leq t$ are truly relevant (also called *hits*), let $r(t) = \frac{h(t)}{\pi}$ and $p(t) = \frac{h(t)}{t}$ be the *recall* and *precision* respectively. Typically, $h(t)$ and $p(t)$ takes values only at a finite number of points $t_i = \frac{i}{n}, i = 1,2,...,n$. The average precision (AP) can then be computed using (4.7):

$$AP = \sum_{i=1}^{n} p(t_i)\Delta r(t_i) \tag{4.7}$$

where $\Delta r(t_i) = r(t_i) - r(t_{i-1})$. Details on the AP criterion can be found in [74].

4.5 Algorithms Explored

This section provides the details of the algorithms used in the experiment and in the results section we provides results of these algorithms and discuss their usefulness in fraud detection.

4.5.1 Logistic Regression

Logistic regression is used as a regression analysis to determine the relationship between continuous or categorical feature variables knows as predictor variable with the binary class outcome of the instance. Logistic regression outputs the probability of falling one instance to one class over the other. The probability is determined using defined equation modeled by training the algorithm with training instances. Specific logistic regression model used in this research is defined in [18] by Fan et al.

4.5.2 Support Vector Machines

Support Vector Machine [15] is a very widely used classification algorithm for binary class classification. Introduced by Vapnik et al in early 1995, SVM implements the idea of mapping feature vectors to higher dimensions to use in classification. Non linear mappings known as kernel functions are used to convert the feature to higher dimensions. Out of large training dataset, few instances are identified as support vectors after the training process. In the training phase on the maps higher dimensional data, simple linear decision boundaries are used to classify. Then these supports vectors are used to classify new instances. Defining suitable kernel function for mapping is an inherent problem in SVM where similar problem

exists in Neural Network for selecting a proper network structure. Usually Train and Error method used to identify proper kernel function which range from Linear, Polynomial, Radial Basis or Sigmoid functions.

4.5.3 Bayesian Networks

Bayesian Networks [25] also known as belief networks belongs to statistical probabilistic models for classification. It used graph modeling where each node in the graphs represents a random variable in the dataset. Edges are used as the dependencies between these random variables. Statistical inference techniques are used to determine the strength of these dependencies. BNs are heavily used in artificial intelligence applications for inference statistical properties.

4.5.4 Multi Layer Perceptron

MLP [22] is a type of Neural Network models that uses multiple layers in the network, thus providing more accuracy when the data is not linearly separable. Widely used in many data mining and machine learning applications, MLP uses gradient descend or other defined training functions to train the network based on the weight adjustments for every layer presented. Defining number of hidden layers and the number of perceptrons in each layer is a crucial issue with MLPs.

4.5.5 Decision Tree Algorithms

C4.5 Algorithm

C4.5 is a most commonly used algorithm to generate decision trees using the concept of information gain based on entropy. C4.5 is the successor to ID3 algorithm by the same author Ross Quinlan. The algorithm works as follows.

Let the training set $S = s_1, s_2 ..... s_n$ be the sample which already know the class labels. Each instance s_i contains the features $x_1, x_,2 .... x_m$ where it defines a m

dimensional vector, x_j represent a feature or attribute.

At each node of the tree C4.5 chooses the attributes of the data that most effectively splits the training dataset into subsets of one class or the other. The splitting criteria is based on the normalized information gain. The feature with the highest information gain is used to make the decision and removed from the candidate features for the next decisions. C4.5 algorithm recursively perform the same steps on the subsets until it achieves a desired output. The pseudo code for the algorithm is as follows.

Check for the base cases;
for *each attribute a* **do**
| Find the normalized information gain from splitting on a
end
Let a_{best} be the attribute with the highest normalized information gain;
Create a decision node that splits on a_{best};
Recurse on the sublists obtained by splitting on a_{best}, and add those nodes as children of node;

Algorithm 1: Psuedo Code for C4.5 Decision Tree Algorithm

RepTree

RepTree is a fast decision tree learner available in WEKA machine learning tool. RepTree also builds the decision tree using information similarly in C4.5 and the decision tree pruning performed using reduced-error pruning technique with backfitting. When it encounters missing values, RepTree dealt with such situations by splitting the corresponding instances in to pieces same as in C4.5. RepTree considers all attributes for decision making processing in every node where similar algorithm *RandomTree* only uses set of K randomly chosen attributes.

RandomForest

In *RandomForest* each tree in the forest is built from a sample drawn with replacement based on bootstap method from the train set [8]. Moreover, when splitting a

node during the tree construction, the split chosen is not the best split among all features, but rather the best among a random subset of the features. Consequently, the bias of the forest usually slightly increases, but its variance also decreases, usually more than compensating for the increase in bias, thus giving an overall better model.

4.6 Experimental Flow

The experiment flow from the raw dataset to final model is explained in this section. We start with analyzing raw data to preprocess to be used in training.

4.6.1 Preprocessing

The raw dataset is analyzed and found that it contains $\tilde{2}$ million click details in each dataset. Thus MySQL server is used to store the data since a database server can handle millions of data at once and can generate ad-hoc queries based on the user requirements. Features are created using SQL queries (Appendix B) and generate the final dataset to use with WEKA and SVM^{light} for training and testing purposes. The Feature Engineering process is explained in the next chapter as it is a core component in our research.

4.6.2 Experimental Environment

The experiment is carried out on the computer system that has following performance as defined in the Table 4.7.

Measure	Discription
Processor	Intel Core -i7 3.40GHz 8CPUs
RAM	16.00 GB
Operating System	Windows 7 Professional 64bit
Database Server	MySQL server 5.2.44

Table 4.7: Experimental environment.

We used GUI and CLI of WEKA in our experiment and output the prediction scores as CSV files to be used in Model Combiner. The steps of experimental procedure is as follows.

1. Load the Data to MySQL server
2. Use SQL queries to derive features
3. Combine the features with labels for Training, Validation and Test sets
4. Train the models with Training set and perform testing on Validation set
5. Based on the test results change the model, re-train and re-evaluate.
6. Final model is saved to be used in model combiner
7. Use model combining approach on different models and evaluate with validation set to produce the final model.

4.6.3 Sampling Techniques

As the dataset is highly imbalanced with respect to class distribution, we used Sampling techniques on the training dataset to experiment the effect on validation and test sets. We use different sampling techniques as SMOTE implementation in WEKA, Resampling in WEKA and our informative undersampling using clustering method.

4.6.4 Informative Undersampling

Undersampling is a widely used technique when class distribution is highly imbalanced, and it selects some instances from the majority class to be used with all the instances from the minority class. In order to select instances from majority class, one can use random technique, but when used that results can be extremely bad.

Thus, information can derive from the dataset and the domain is used to select such instances. In our experiment we use clustering methods to select the instances. The procedure is as follows.

1. Exclude the minority class (Frauds) from the training dataset
2. Use clustering technique (k-means) to cluster the majority class based on the desired class distribution
3. Select the cluster heads from each of the clusters to be used with minority class examples in the training set.

For example, in our training dataset only 72 Frauds presents with compared to 3009 legitimate (OK) instances. When we need to derive a training sample which is balances class distribution we exclude all 72 frauds from the training set. Use rest 3009 instances with clustering algorithm with number of clusters as 72 which is equal to number of fraud instances. Then cluster head from each 72 clusters are selected. Thus final training set has 72 Fraud instances and 72 OK instances selected as cluster heads. In our experiment we used following sampled datasets to evaluate models.

- Cluster Sampling with Fraud:OK distribution as 50:50
- Cluster Sampling with Fraud:OK distribution as 33:67
- Cluster Sampling with Fraud:OK distribution as 15:85
- SMOTE oversampling producing 100% SMOTE instances
- SMOTE oversampling producing 500% SMOTE instances
- Resampling in Weka - Fraud:OK distribution to 30:70

Results on these sampled training sets as well as full dataset with algorithms defined in this section is explained in the Chapter 6.

CHAPTER 5

Feature Engineering

5.1 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, partnerid, campaign ID, country, time-at, category, referrer in "click" dataset and partnerid, 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. These attributes can be null and/or not associated with any publisher behaviors.

After selecting the attributes which were to be considered for further consideration, the *trainingPublisher* data set was merged with the *traningClick* data set

to have the status of each publisher/partner in click dataset. Each publisher/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.

5.2 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 set 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 [44, 59, 35] and identify some features that have been used in the Google adsense fraud detection mechanism and other PPC fraud detection systems. 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 mins, 1 hour, 3 hours 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 Figure 5.1 shows all the features we derive from one single attribute ‘time-at’ from given original dataset.

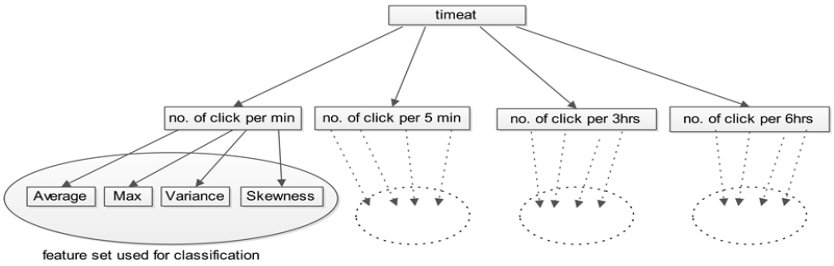


Figure 5.1: Feature creation from the "time-at" attribute.

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.

1. MaxSameAgentClicks = for a given partner, count the number of clicks from different Agents and then obtain the maximum of that.

2. MaxSameAgentClicks = for a given partner, count the number of clicks from different Agents and then obtain the Average of that.

3. VarSameAgentClicks = for a given partner, count the number of clicks from different Agents and then obtain the Variance of that.

4. SkewnessSameAgent Clicks = for a given partner, count the number of clicks from different Agents and then obtain the Skewness of that.

Category

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.

Country

Similarly we define features on Country where we calculate clicks from unique countries, their variance and entropy.

Campaign ID

For all unique Campaign IDs used to generate features such as the number of unique campaigns for each publisher, variance etc.

Referrer

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 40 different features created from different individual and set of attributes from the data set. The list of those 40 features is provided in Appendix A. These carefully crafted features always have threat of over fitting the model, which can be solved by feature selection

method. The sample SQL queries that used to generate these features are indicated in Appendix B.

5.3 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 [39] resulted in the highest accuracy compared to the other methods for feature selection. But when using full feature set it provides higher accuracy, thus feature selection is excluded from the final model.

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.

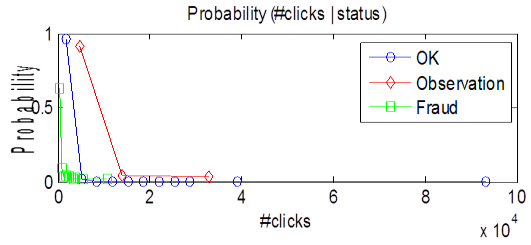
CHAPTER 6

Results and Analysis

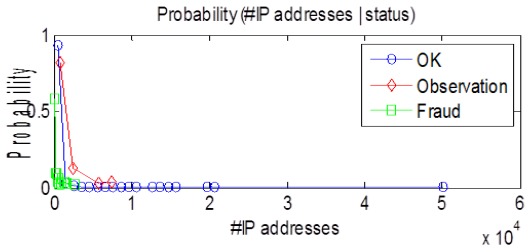
Our approach to detecting fraud consists of employing contemporary classification models over data derived from click database. We tuned the parameters of these models such that they work robustly with the train and validation datasets, and can subsequently generalize to any unseen test set. We tried a range of different model parameters that yielded the highest precision, recall, F-Measure and area under the receiver operating characteristics (ROC) curve, with low standard deviation to ensure performance consistency. This chapter explains the results obtained using different methods and algorithms.

We screen through the initial dataset generated with respect to classes. The four features with respect to the classes are depicted in Figure 6.1. Interestingly we can see that fraud publishers have a low click probability compared to other two OK and Observation Classes. This implies that the number of clicks generated for fraudulent publishers is less compared to OK and Observation publishers. Fraudulent publishers don't want to be noticed, thus they generate adequate number of

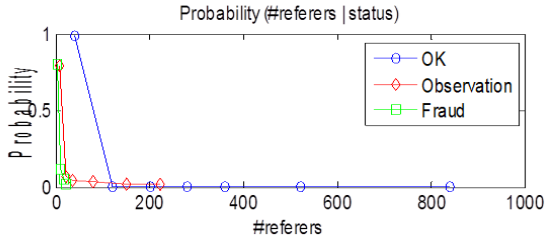
clicks to hide among the legitimate publishers.



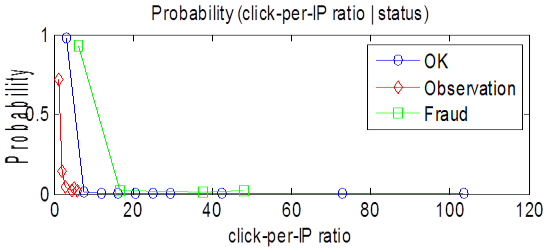
(a)



(b)



(c)



(d)

Figure 6.1: Distribution of the train dataset with the status: (a) Number of clicks, (b) Number of unique visitors, (c) Number of unique referer, and (d) Click per visitor ratio (number of clicks divided by number of unique visitors).

On the other hand, we found that in terms of referer distribution, the Fraud

publishers are rather different from the OK publishers and have low probability similar to the Observation publishers. Further investigation revealed that many Fraud publishers have empty/unknown referer fields. Hence, features derived from referer can be good indicators for fraudulent acts. Lastly, the distribution of the click per visitor ratio in Figure 6.1(d) shows that Fraud publishers have higher ratio than the other groups, suggesting that the former focus on more efficient use of resources (IP address, in this case) to inflate the click traffic. This motivates us to investigate several other ratio-based features such as click per referer ratio, click per agent ratio, click per country ratio, etc.

6.1 Single Algorithms

6.1.1 Conventional Classification Learners

In order to testify our proposed method, we perform training and validation using single algorithms and analyze the performance of each algorithm with respect to performance metrics. We use simple logistic regression with ridge parameter of 1.0E-8. We also used SVM using different kernels to train and validate the data and the resulting prediction performances are indicated in Table 6.1.

Algorithm	Precision		Recall		F-Measure		ROC	
	OK	Fraud	OK	Fraud	OK	Fraud	OK	Fraud
Logistic Regression	0.98	0.543	0.993	0.294	0.986	0.382	0.799	0.799
Bayesian Network	0.996	0.116	0.805	0.894	0.89	0.205	0.863	0.864
SVM	0.982	0.026	0.298	0.778	0.457	0.05	0.538	0.538
Multilayer Perceptron	0.976	0.522	0.996	0.114	0.986	0.222	0.767	0.767

Table 6.1: Performance measures with single algorithms.

As shown in Table 6.1 when using single classifier, the results of the model are not to a satisfactory level which it depicts best performance with Simple Logistic Regression and still it accounts to 0.382 for F-Measure, and 0.799 for AUC of ROC

Curve with respect to Fraud Class.

6.1.2 With Ensemble Approach

Since a single algorithm does not produce results with satisfying performance we use Ensemble Learning Approach to deal with current imbalance dataset. We use both bagging and boosting methods as ensemble methods and carried out experiment with different weak learner algorithms. The resulting model performances are indicated in Table AA.

Algorithm	Precision		Recall		F-Measure		ROC	
	OK	Fraud	OK	Fraud	OK	Fraud	OK	Fraud
Bagging J48	0.982	0.721	0.996	0.365	0.989	0.484	0.936	0.936
Bagging Random Forest	0.978	0.667	0.997	0.212	0.987	0.321	0.932	0.933
Bagging RepTree	0.975	0.667	0.998	0.118	0.987	0.2	0.934	0.934
Adaboost J48	0.98	0.568	0.994	0.294	0.987	0.388	0.865	0.893
Adaboost Random Forest	0.978	0.75	0.998	0.212	0.988	0.33	0.725	0.825
Adaboost RepTree	0.98	0.393	0.988	0.282	0.984	0.329	0.894	0.894
Stacking NB<J48<MLP	0.986	0.319	0.969	0.506	0.977	0.391	0.855	0.854
Stacking NB<BJ48< BRF	0.98	0.491	0.991	0.306	0.986	0.377	0.847	0.847

Table 6.2: Performance measures with ensemble algorithms.

As shown in the Table 6.2, J48, which is the WEKA implementation of C4.5 algorithm when used with bagging as an ensemble method, produced the most accurate results compared to the other models. Bagging with Rep Tree perform badly having 0.2 as F-Measure but it has higher AUC of ROC compared to boosting methods.

We use the Adaboost ensemble method with same Decision tree algorithms and Adaboost with J48 provides better results compared to other decision tree algorithms used with Adaboost.

We use performance measure Average precision to evaluate these methods and results are depicted in Figure 6.2. Average precision works on the prediction ranking rather than actual number of prediction to OK and Fraud classes. Thus, if one

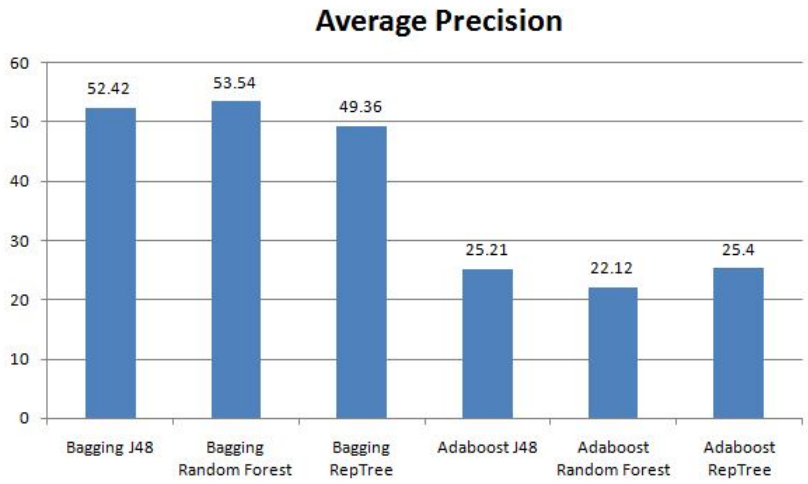


Figure 6.2: Average precision of the ensemble learning algorithms.

algorithm classify more fraud instances as frauds with few OK instances as frauds, but if these missclassified instances ranked higher than actual fraud instances, then overall performance can be lower. Similar situation happens with above mentioned methods. As shown in the figure higher average precision acheived by Bagging with Random Forest followed by Bagging with J48 algorithms. Boosting algorithms that work better than Bagging Random Forest and Bagging Rep Tree with respect to F-measure, perform badly with respect to Average Precision achieving 25.4% for Adaboost with Rep Tree which ranked high in the Average Precision measure.

6.1.3 Sampling + Ensemble Approach

As mentioned in the previous sections the fraud detection data is very imbalanced with respect to the class distribution. Thus, we used sampling techniques on the data set and used resampled data for training and the trained model then used on full validation set for prediction. We use random undersampling, SMOTE over-

sampling, random oversampling and the informative undersampling. For the informative undersampling we used k-means clustering on the data set and selects the cluster-heads as the sampled instances. The respective performance values with these datasets are mentioned in Table 6.3. We use the best ensemble algorithms obtained from the previous sections to use with sampled data. Thus we use Bagging + J48 and Bagging + Random Forest with sampled dataset.

Algorithm	Precision		Recall		F-Measure		ROC	
	OK	Fraud	OK	Fraud	OK	Fraud	OK	Fraud
Clustering 15:85	0.987	0.425	0.978	0.565	0.983	0.485	0.917	0.917
Clustering 50:50	0.996	0.133	0.836	0.882	0.909	0.231	0.911	0.911
Clustering 33:67	0.994	0.184	0.896	0.824	0.942	0.3	0.914	0.914
SMOTE 100% Instances	0.982	0.516	0.99	0.376	0.986	0.435	0.917	0.917
SMOTE 500% Instances	0.983	0.388	0.983	0.388	0.986	0.388	0.916	0.916
Resampling 30:70	0.984	0.358	0.977	0.459	0.98	0.402	0.794	0.794

Table 6.3: Performance measures with sampling and ensemble algorithms.

We evaluate trained model with sample data on the validation set with respect to Average Precision and depicted in Figure 6.3. When using SMOTE to oversample the minority class generating 100% synthetic instances, the results with respect to Average Precision shows higher accuracy achieving 45,26% and Cluster Sampling to class distribution 15:85 for Fraud:OK has second highest accuracy. But the latter has higher accuracy with respect to F-Measure.

6.1.4 Cost Based Algorithms

We evaluate training and validation set on the cost based learning methods. Cost based learning is widely used as minority class is most important than majority class. We use cost parameters as missclassifying a Fraud instance as OK instance has 10 times higher cost than misclassifying a OK instance as a fraud instance. We also use equal weights with the algorithm to analyze the performance and results are shown in Table 6.4.

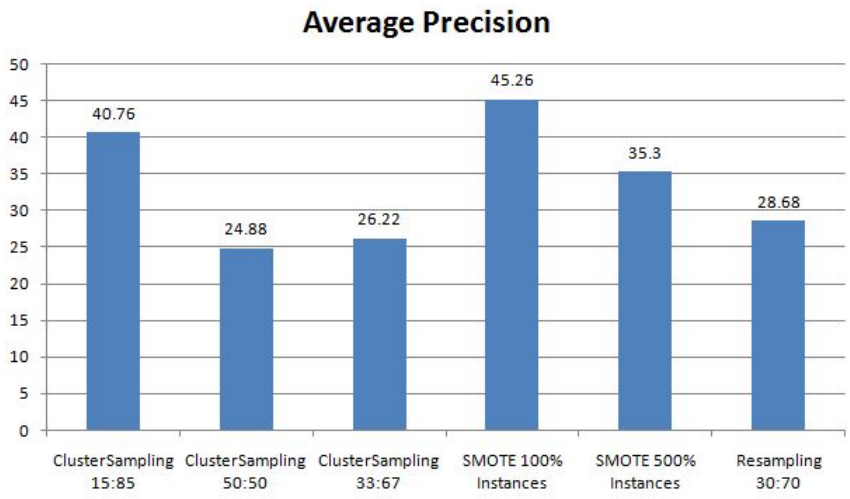


Figure 6.3: Average precision of validation set with the sampled data + ensemble algorithm J48.

Algorithm	Precision		Recall		F-Measure		ROC	
	OK	Fraud	OK	Fraud	OK	Fraud	OK	Fraud
MetaCost BaggingRandomForest 10:1 for Fraud:OK	0.986	0.303	0.967	0.506	0.976	0.379	0.785	0.785
MetaCost BaggingRandomForest 1:1 for Fraud:OK	0.981	0.458	0.989	0.318	0.985	0.375	0.765	0.765
CostbasedLearning 10:1 for Fraud:OK	0.981	0.483	0.99	0.341	0.985	0.4	0.842	0.842

Table 6.4: Performance measures with cost based learning with ensemble learner as Bagging + Random Forest.

The respective confusion matrices obtained on the validation set also shown in Table 6.5. When used with 10:1 cost for Fraud:OK it tends to detect more frauds, but also misclassify more OK instances as Fraud.

6.2 Combining Multiple Algorithms

After careful analysis of results obtained with single algorithms we further investigates other methods to improve the results when presents imbalance class distribution. Phua et al presents combining of multiple models for improved results [55]

	OK	Fraud
OK	2880	99
Fraud	42	43

(a)

	OK	Fraud
OK	2947	32
Fraud	58	27

(b)

	OK	Fraud
OK	2948	31
Fraud	56	29

(c)

Table 6.5: Confusion matrix of MetaCost with 10:1 (a) MetaCost with 1:1 (b) and cost based learning with 10:1 weights for Fraud:OK with Bagging + Random Forest.

when minority class has very few instances.

For model combining we first examine the effect of combining ensemble methods trained with tree based algorithms. Thus, we use simple averaging over the predicted confidence values for all models. After averaging over the confidence value for Bagging + J48, Bagging + Random Forest, Bagging + RepTree, Adaboost + J48, Adaboost + Random Forest, Adaboost + RepTree the results shown in the Table 6.6 for Confusion Matrix and 6.7 for performance measures are obtained. Interestingly after combining the models, the average precision for validation set was 53.48% which is less than the Bagging with Random Forest model itself.

	OK	Fraud
OK	2962	17
Fraud	57	28

Table 6.6: Confusion matrix for combined model of ensemble method Bagging and Boosting with J48, Random Forest and RepTree.

Precision		Recall		F-Measure	
OK	Fraud	OK	Fraud	OK	Fraud
0.9943	0.3294	0.9811	0.6222	0.9877	0.4308

Table 6.7: Performance values for combined model of ensemble method Bagging and Boosting with J48, Random Forest and RepTree.

Since ensemble method with single Decision Tree algorithms or combined model does not produce good results we examine combining with Sampling methods. Thus we evaluate many combinations from the decision tree algorithms with bagging and boosting methods with the defined sampling methods. With the differ-

ent combinations tried we found that following combination provides acceptable accuracy with the performance values shown in Table 6.9. Respective confusion matrix is defined in Table 6.8.

	OK	**Fraud**
OK	2948	31
Fraud	49	36

Table 6.8: Confusion matrix for combined model of Bagging J48, Bagging Random Forest, Metacost with Bagging J48 with cost 10:1, Bagging + J48 with 100% Smote instances and Clustering-based sampling for Fraud-15:85-OK.

Precision		Recall		F-Measure	
OK	Fraud	OK	Fraud	OK	Fraud
0.9896	0.4236	0.9837	0.5374	0.9867	0.4737

Table 6.9: Performance values for combined model of Bagging J48, Bagging Random Forest, Metacost with Bagging J48 with cost 10:1, Bagging + J48 with 100% Smote instances and Clustering-based sampling for Fraud-15:85-OK.

We further investigate any possible methods to optimize the results for higher accuracy. It is found that *Random Sub Space* used with *J48* when combined with other models defined early provides better results. Thus, we examine combined models with Random Sub Space + J48 with other ensemble learning methods and sampling methods.

After step by step improvements to the model, we finally reach a model that can generalize and can be used to detect *Fraud* instances more accurately. The final model used is a combination of following models. Simple averaging is used as a combination approach.

Bagging + J48, Random Sub Space + J48, Bagging + J48 + Clustered Sample Fraud-33:67-OK, Bagging + J48 + Clustered Sample Fraud-15:85-OK

The results obtained using above model on validation set is shown in Table 6.11 for confusion matrix and Table 6.10 for performance measures.

After evaluating on validation set and define the final model we perform exper-

	OK	Fraud
OK	2935	44
Fraud	39	46

Table 6.10: Performance values for the final combined model of Bagging J48, Random Sub Space with J48,Clustering Based sampling for Fraud-33:67-OK, Clustering-based sampling for Fraud-15:85-OK.

Precision		Recall		F-Measure	
OK	Fraud	OK	Fraud	OK	Fraud
0.9852	0.5412	0.9869	0.5111	0.9860	0.5257

Table 6.11: Confusion Matrix for the final combined model of Bagging J48, Random Sub Space with J48,Clustering Based sampling for Fraud-33:67-OK, Clustering-based sampling for Fraud-15:85-OK.

iment of Test set as well.

6.3 Evaluation on Test Set

Since the single algorithm as defined in the previous section do not perform well with the imbalance dataset we have, we used other methods to evaluate Test Set. Thus, first we test the models in Table 6.2 with Test Set and the results are as follows. Table 6.12 depicts the confusion matrix for each algorithm and trained with bagging method and Table 6.13 depicts confusion matrix for Boosting method. Performance measure for all 6 algorithms are in Table 6.14.

	Bagging J48		**Bagging RF**		**Bagging RepTree**	
	OK	**Fraud**	**OK**	**Fraud**	**OK**	**Fraud**
OK	2896	22	2908	10	2915	3
Fraud	61	21	69	13	75	7
	a		b		c	

Table 6.12: Confusion matrix for bagging ensemble algorithms on test set.

We also perform the evaluation on models that trained with sampling methods. We use models that use SMOTE sampling as well as our cluster sampling method. The results are depicted in Table 6.15 for performance measures. Respective con-

	Adaboost J48		Adaboost RF		Adaboost RepTree	
	OK	**Fraud**	**OK**	**Fraud**	**OK**	**Fraud**
OK	2904	14	2912	6	2872	46
Fraud	66	16	69	13	56	26
	a		b		c	

Table 6.13: Confusion matrix for Boosting ensemble algorithms on test set.

Algorithm	Precision		Recall		F-Measure		ROC	
	OK	Fraud	OK	Fraud	OK	Fraud	OK	Fraud
Bagging J48	0.979	0.488	0.992	0.256	0.986	0.336	0.948	0.948
Bagging Random Forest	0.977	0.565	0.997	0.159	0.987	0.248	0.969	0.969
Bagging RepTree	0.975	0.7	0.999	0.085	0.987	0.152	0.97	0.97
Adaboost J48	0.978	0.533	0.995	0.195	0.986	0.286	0.859	0.908
Adaboost Random Forest	0.977	0.684	0.998	0.159	0.987	0.257	0.666	0.775
Adaboost RepTree	0.981	0.361	0.984	0.317	0.983	0.338	0.886	0.886

Table 6.14: Performance measures with ensemble learners on test set.

fusion matrices are in Table 6.16 and Table 6.17.

Algorithm	Precision		Recall		F-Measure		ROC	
	OK	Fraud	OK	Fraud	OK	Fraud	OK	Fraud
Cluster Sample 50:50	0.998	0.143	0.842	0.939	0.914	0.249	0.945	0.945
Cluster Sample 33:67	0.997	0.199	0.9	0.89	0.946	0.326	0.955	0.955
Cluster Sample 15:85	0.982	0.337	0.98	0.354	0.981	0.345	0.952	0.951
Smote 100% Instances	0.98	0.5	0.992	0.293	0.986	0.369	0.958	0.958
Smote 500% Instances	0.981	0.371	0.985	0.317	0.983	0.342	0.952	0.952

Table 6.15: Performance measures with sampling with ensemble learner on test set.

With the results based on the Training and Validation we select the best model as the simple averaging of algorithms specified in 6.10. Thus, we evaluate our final model on the Test set and the results are as follows. Confusion Matrix on Test Data with the final model is given in Table 6.18.

Corresponding performance measures are listed in Table 6.19. From the data we can see that the combined model achieved higher accuracy also on Test Set with F-measure accounts to 0.4432 for Fraud Class. Thus our model can be generalize to detect many frauds in very imbalance data set. More discussion on the results

	Cluster Sample 50:50		Cluster Sample 33:66		Cluster Sample 15:85	
	OK	**Fraud**	**OK**	**Fraud**	**OK**	**Fraud**
OK	2458	460	2625	293	2861	57
Fraud	5	77	9	73	53	29
	(a)		(b)		(c)	

Table 6.16: Confusion matrix for Cluster-based Sampling with Bagging + J48.

	Smote 100% Instances		Smote 500% Instances	
	OK	**Fraud**	**OK**	**Fraud**
OK	2894	24	2874	44
Fraud	58	24	56	26
	(a)		(b)	

Table 6.17: Confusion matrix for SMOTE-based sampling with Bagging + J48.

	OK	**Fraud**
OK	2863	55
Fraud	43	39

Table 6.18: Confusion Matrix for Test Set on Final Model.

Precision		**Recall**		**F-Measure**	
OK	**Fraud**	**OK**	**Fraud**	**OK**	**Fraud**
0.9812	0.4756	0.9852	0.4149	0.9832	0.4432

Table 6.19: Performance measures for test set on the final model.

and possible improvements are described in the next chapter.

CHAPTER 7

Discussion and Future Work

In this research we presented a combined approach to cope with Imbalance Class Distribution problem using both ensemble method and sampling method. Our analysis shows that when using ensemble method or sampling method individually, it does not produce very accurate results with respect to interesting minority class. It also shows that traditional machine learning algorithms do not perform well on the very imbalanced data sets. We can define the order of algorithms in increasing order of accuracy with respect to the minority class as follows.

1. Traditional Basic Classifiers
    - SVM, Naïve Base, Multi-Layer Perceptron, Logistic Regression
2. Ensemble Learners
    - Bagging with C4.5(J48), Bagging with Random Forest, Adaboost with C4.5

3. Sampling with Ensemble Learners

    - SMOTE with Bagging + C4.5, Cluster Based Sampling with Bagging + C4.5

4. Combining Multiple Algorithms with Ensemble and Sampling Methods

    - Bagging with C4.5 + Cluster Based Sampling to 15:85 with Bagging C4.5

Thus, based on our results, we can argue that when imbalance data set is used, sampling techniques together with ensemble learners can be used for higher accuracy.

7.1 Sampling Technique

Our experiment shed lights to a sampling method which can be generalized to use with other class imbalance problems. As random undersampling failed to produce good results we used informative undersampling. Thus, we used clustering methods to cluster the majority class according to the number of instances we need. Select the cluster heads as the instances and perform ensemble learning on the sample selected from this method. It is proved to produce higher results when using two such clusters having high percentage for minority class and low percentage for minority class together with other ensemble learning algorithms which trained using full training data set. We believe the use of two clustered samples increase the information about majority class which does not present when select individual clustering based samples.

7.2 Insights from Feature Creation

Another main contribution from this research is in feature engineering where we define multiple features for seven attributes presents in the click database. After careful analysis of attributes and generated features, we found out that some features are really crucial for detecting click frauds. These include time series features such as click ratio in five minutes intervals, three hours intervals as well as spatial features such as ip-click ratio, referrer-click ratio etc. When used with feature selection methods, it does not produce good results, which we can conclude that the current feature are necessary for proper classification and creating more features might increase the accuracy.

7.3 Comparison with Similar Methods

The Singapore Management University organized a competition on Fraud Detection in Mobile Advertising in conjunction with 4th Asia Conference in Machine Learning. As we participated in the competition, we eventually had a platform to compare our results with other competitors in terms of the requirements in the competition. The requirement of the competition was to implement a model that can classify fraud publishers from legitimate publishers achieving higher Average Precision. Thus, we compare our model with Phua et al [56], Wei [67], Berrer [7] and Shivashankar et al [60].

The model we used in the competition consists of averaging confidence values for ensemble learning methods [54], and it ranked 2nd among the competitors. The model proposed by Phua et al winning the 1st place, and ranked 4th on Validation set provides insights on using ensemble approach on carefully crafted features. In their model, they used 108 features generated from seven attributes provided. On validation set they achieve 59.38% average precision where we man-

age to achieve 59.39%. But on the test set they achieve 51.54% while we achieve 46.41%. To build the model, authors used Generalized Boosted Regression Model in ”R” Package. As the only evaluation parameter is Average Precision, we cannot simply compare our new results with the existing models proposed by the competitors.

Wei as the 3rd winner in the competition used a blend of Decision Tree models and Neural Network model to design the fraud detection model and used 30 features without feature selection and 17 features after using backward feature elimination. The final model is a combination of 10 models consisting of decision trees, BaseNet classifiers and neural network based classifiers. Berrer and Shivasankai in their models also used a combination of ensemble learners to achieve higher results on the provided imbalanced dataset. The numbers of features created in both models are less than our current model.

As the current model perform better than the model submitted for the competition in terms of detecting fraud instances, in real life scenario we can use the model proposed in this research. This is due to ranking of the detected in not important but the number of detections is crucial in real life scenario.

7.4 Further Optimizing

Many applications use two step methods to verify the authenticity, which we can adopt to current model. The current model has 1.88% False Positive rate which we can reduce by using two step verification. It is a common practice of the advertising company to use more than one click to reach the actual companys website. But when employing this model, due to its inherent feature, company might lose some portion of potential customers. But without putting this 2nd click for every ad for every publisher, based on the model prediction, we can place the 2nd link for the

ads deleivered to publishers classified as fraud. Clickbots used by fraud publishers are found to be ineffective with the 2nd click, if the link is obfuscated. As we place the 2nd link only on few ad publishers, the amount of affected customers are two few which will not harm for the companys business.

7.5 Future Work

Experiments conducted on imbalanced data set with model combining approach opens new avenues which we going to explore to further optimize our results.

For feature extraction, we focused on individual attributes rather than combination of multiple attributes. But it is found out that features that can be generated using multiple attributes might produce more results. Such example is the number of duplicate clicks by each publisher. Fraudulent publishers since they use clickbots tend to generate more duplicate clicks compared to legitimate publishers. Also the number of clicks generated in certain time of the day might important as fraud publishers use computers in working time rather than free time which legitimate users used to surf the internet. As such, we are focusing on more feature engineering techniques to employ with our model for more accurate results.

Informative undersampling using clustering based methods tend to produce better results compared to random sampling as well as SMOTE sampling. Thus, better clustering methods are yet to evaluate as we focus on using k-means clustering on this experiment. Clustering methods which can capture more information with less number of instances (clusters) are needed to evaluate which yield to obtain a more balance dataset which has same information as in the imbalance dataset.

Better model combining method than simple averaging could produce better results when used with more models. Thus, as future work we will examine the effect of model combining algorithm which range from simple averaging, majority

voting, averaging on majority voting etc. More models will be used to evaluate the effect on full training set as well as reduced sampled dataset.

The conclusions we can made from this experiment is described in the next chapter.

CHAPTER 8

Conclusion

Fraudsters can be found in any business model, but the severity of the attacks ranging from negligible to catastrophic. Energy networks that will be deployed in the near future will suffer more sophisticated fraudulent attacks as current conventional energy network suffers. It is necessary to develop more accurate fraud detection techniques cope with such incidents as the effects of these attacks can be catastrophic.

Similar to smart grid networks, conventional online advertising networks suffer fraud attacks, thus analyzing these attacks and come up with generic solutions will shed the lights to develop fraud detection system for smart grid as well as any other business model that affects by fraudsters. Thus, in our research we analyzed click data on mobile advertising network, to design a fraud detection system.

The profound factor of any system that vulnerable to fraudsters is that, only a small percentage of fraudulent behaviors exist, but detecting this small portion is crucial for smooth run of the business. This implies an imbalance class distribution

problem for any detection system that wants to develop.

We analyze the effect of single use of algorithms ranging from simple classification algorithm to ensemble learning algorithms such as bagging and boosting. Based on our results we can conclude that conventional classification algorithm under perform on the highly imbalanced dataset, whereas ensemble learning algorithms tend to produce better results. It is also analyze the effect of sampling methods to under sample or over sample the dataset. Results indicate our informative under sampling method outperform SMOTE over sampling on the given dataset. Our experiment shows that combining multiple algorithms on ensemble learning and informative under sampling with proper model combining approach can achieve higher results. It is also shows that when using cluster based informative sampling, it is necessary to achieve as many information as possible with few number of data samples. Thus, it is necessary to have at least two samples, one with higher percentage of minority class instances and one with lower percentage of minority class instances.

Feature engineering plays a critical role in any classification and it is even critical when class imbalance problem exists. Thus, devising more features from the dataset can improve the quality of the results and it is tested and proved in our experiment. A carefully crafted feature set along with proper blend of ensemble learning algorithm and informative sampling method could yield to a higher accuracy in classification problem when there exists imbalance class distribution.

APPENDIX A

Full Feature Set Used

The following table shows the full feature set and description about those features.

Feature	Description
MaxSameAgentCount	Maximum number of clicks received from a single agent for each publisher
MaxSameCidCount	Maximum number of clicks received from a single Campaign ID for each publisher
TotalClicks	Total number of clicks received by each publisher
MaxSameCntrCount	Maximum number of clicks received from a single country for each publisher
MaxSameIPCount	Maximum number of clicks received from a single IP for each publisher

Table A.1: Full feature set used in the experiment - Part 1

Feature	Description
AvgClickPerMin	Average number clicks received by each publisher in 1 min time intervals
Varperminclick	Variance of the clicks received by each publisher in 1 min time intervals
AvgClicksPer5Min	Average number clicks received by each publisher in 5 min time intervals
MaxClicksPer5Min	Maximum number clicks received by each publisher in 5 min time intervals
Varper5minclick	Variance of clicks received by each publisher in 5 min time intervals
MaxClicksperMin	Maximum number clicks received by each publisher in 1 min time intervals
AvgClicksPer3hrs	Average number clicks received by each publisher in 3 hour time intervals
MaxClicksPer3hrs	Maximum number clicks received by each publisher in 3 hour time intervals
Varper3hrsclick	Variance of clicks received by each publisher in 3 hour time intervals
AvgClicksPer6hrs	Average number clicks received by each publisher in 6 hour time intervals
MaxClicksPer6hrs	Maximum number clicks received by each publisher in 6 hour time intervals
Varper6hrsclick	Variance of clicks received by each publisher in 6 hour time intervals
NoOfAgents	Total number of agents which clicks received by each publisher
Click/AgentRatio	Average click ratio for each agent
NoOfCid	Total number of campaign IDs which clicks received by each publisher
Cid/ClickRatio	Average click ratio for each Campaign
NoOfCountries	Total number of countries which clicks received by each publisher
Country/ClickRatio	Average click ratio for each country
NoOfIPs	Total number of unique IPs which clicks received by each publisher
IP/ClickRatio	Average click ratio for each IP
NoOfRefers	Total number of referrers which clicks received by each publisher
Refer/ClickRatio	Average click ratio for each Referrer
nonreferedclickratio	Total number of non referrerd click with respect to total clicks received for each publisher

Table A.2: Full feature set used in the experiment - Part 2

Feature	Description
CategoryPrior	Prior Probablity of being fraud in each category
AgentEntropy	Entropy of the click distribution on Agent Attribute
CidEntropy	Entropy of the click distribution on Campaign Attribute
CountryEntropy	Entropy of the click distribution on Country Attribute
AgentVar	Variance of the clicks for each Agent for each publisher
IPVar	Variance of the clicks for each IP for each publisher
cntrVar	Variance of the clicks for each Country for each publisher
campVar	Variance of the clicks for each Campaign for each publisher
skew1min	Skewness on the clicks received in 1 min intervals for each publisher
skew5min	Skewness on the clicks received in 5 min intervals for each publisher
skew1hr	Skewness on the clicks received in 1 hour intervals for each publisher
IPEntropy	Entropy of the click distribution on IP Attribute

Table A.3: Full feature set used in the experiment - Part 3

APPENDIX B

Sample SQL Queries

Sample SQL queries used to generate features from combined click database and publisher (partner) database.

Feature	Query
Maximum Clicks in 5 Mins	select a.pid,MAX(a.cpid),a.status from (SELECT timeat, partnerid as pid, count(partnerid) as cpid, status FROM test GROUP BY UNIXTIMESTAMP(timeat) DIV 300, partnerid) as a group by a.pid
Total Clicks for each Publisher	select partnerid,count(partnerid) from clickvalidation group by partnerid
Non Referrered Clicks	select a.pid1,ifnull(b.cnt,0) from (select distinct partnerid as pid1 from test) as a left join (select partnerid as pid2, count(partnerid) as cnt from test where referer = '' group by partnerid) as b on a.pid1=b.pid2 order by a.pid1
Maximum Same Agent Count	SELECT agentCounts.partnerid, MAX(Count) AS MaxCount FROM (SELECT partnerid, agent, COUNT(agent) AS Count FROM clickvalidation GROUP BY partnerid,agent) AS agentCounts GROUP BY partnerid
Category Prior	select partnerid,count(distinct category),count(partnerid),count(partnerid)/count(distinct category) from clickvalidation group by partnerid

Table B.1: Sample SQL queries.

APPENDIX C

Abbreviations

PPC Pay Per Click

PC Personal Computer

SVM Support Vector Machines

ANN Artificial Neural Networks

SMOTE Synthetic Minority Oversampline Technique

CSV Comma Separated Version

URL Unique Resource Locator

TP True Positive

FP False Positive

FN False Negative

TN True Negative

ROC Receiver Operating Characteristic

AUC Area Under the Curve

MLP Multi Layer Perceptron

Bibliography

[1] M. Artis, M. Ayuso, and M. Guillen. Modelling different types of automobile insurance fraud behaviour in the Spanish market. *Insurance Mathematics and Economics*, 24(1-2):67–81, 1999.

[2] Z. Aung, M. Toukhy, J. Williams, A. Sanchez, and S. Herrero. Towards accurate electricity load forecasting in smart grids. In *Proceedings of the 4th International Conference on Advances in Databases, Knowledge, and Data Applications (DBKDA)*, pages 51–57, 2012.

[3] E. Belhadji, G. Dionne, and F. Tarkhani. A model for the detection of insurance fraud. *The Geneva Papers on Risk and Insurance*, 25:517–538, 2000.

[4] T. Bell and J. Carcello. A decision aid for assessing the likelihood of fraudulent financial reporting. *Auditing: A Journal of Practice and Theory*, 10: 169–184, 2000.

[5] M. Beneish. Detecting GAAP violation: Implications for assessing earnings management among firms with extreme financial performance. *Journal of Accounting and Public Policy*, 16:271–309, 1997.

[6] P. Bentley. "evolutionary, my dear Watson". investigating committee-based evolution of fuzzy rules for the detection of suspicious insurance claims. In *Proceedings of the 2000 Genetic and Evolutionary Computing Conference (GECCO)*, pages 701–709, 2000.

[7] D. Berrar. Random forests for the detection of click fraud in online mobile advertising. In *Proceedings of the 2012 Workshop on Fraud Detection in Mobile Advertising (FDMA)*, 2012.

[8] L. Breiman. Random forests. *Machine Learning*, 45:5–32, 2001.

[9] P. Brockett, R. Derrig, L. Golden, A. Levine, and M. Alpert. Fraud classification using principal component analysis of RIDITs. *The Journal of Risk and Insurance*, 69:341–371, 2002.

[10] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer. SMOTE: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, 16:321–357, 2002.

[11] N. V. Chawla et al. Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, 16:321–357, 2002.

[12] R. C. Chen, M. L. Chiu, Y. L. Huang, and L. T. Chen. Detecting credit card fraud by using questionaire-responded transaction model based on support vector machines. In *Proceedings of the 2004 International Conference on Intelligent Data Engineering and Automated Learning (IDEAL)*, pages 800–806, 2004.

[13] O. Chertov, V. Malchykov, and D. Pavlov. Non-dyadic wavelets for detection of some click-fraud attacks. In *Proceedings of the 2010 International Conference on Signals and Electronic Systems (ICSES)*, pages 401–404, 2010.

[14] C. C. Chiu and C. Y. Tsai. A web services-based collaborative scheme for credit card fraud detection. In *Proceedings of the 2004 IEEE International Conference on e-Technology, e-Commerce and e-Service (EEE)*, pages 177–181, 2004.

[15] C. Cortes and V. Vapnik. Support-vector networks. *Machine Learning*, 20: 273–297, 1995.

[16] A. Estabrooks. *A Combination Scheme for Inductive Learning from Imbalanced Data Sets*. DalTech, 2000. URL http://books.google.ae/books?id=rV9gNwAACAAJ.

[17] M. A. Faisal, Z. Aung, J. Williams, and A. Sanchez. Securing advanced metering infrastructure using intrusion detection system with data stream mining. In *Proceedings of the 2012 Pacific Asia Workshop on Intelligence and Security Informatics (PAISI)*, pages 96–111, 2012.

[18] R. E. Fan, K. W. Chang, C. J. Hsieh, X. R. Wang, and C. J. Lin. LIBLINEAR: A library for large linear classification. *Journal of Machine Learning Research*, 9:1871–1874, 2008.

[19] W. Fan. Systematic data selection to mine concept-drifting data streams. In *Proceedings of the 2004 ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 128–137, 2004.

[20] W. Fan, S. J. Stolfo, J. Zhang, and P. K. Chan. Adacost: Misclassification cost-sensitive boosting. In *Proceedings of the 16th International Conference on Machine Learning (ICML)*, pages 97–105, 1999.

[21] K. Fanning, K. Cogger, and R. Srivastava. Detection of management fraud: A neural network approach. *International Journal of Intelligent Systems in Accounting, Finance and Management*, 4:113–126, 1995.

[22] G. W. Flake. Square unit augmented, radially extended, multilayer perceptrons. In *Neural Networks: Tricks of the Trade*, pages 145–163, 1998.

[23] D. Foster and R. Stine. Variable selection in data mining: Building a predictive model for bankruptcy. *Journal of American Statistical Association*, 99: 303–313, 2004.

[24] Y. Freund and R. E. Schapire. Experiments with a new boosting algorithm. In *Proceedings of the 13th International Conference on Machine Learning (ICML)*, pages 148–156, 1996.

[25] N. Friedman, D. Geiger, and M. Goldszmidt. Bayesian network classifiers. *Machine Learning*, 29:131–163, 1997.

[26] G. Fumera, R. Fabio, and S. Alessandra. A theoretical analysis of bagging as a linear combination of classifiers. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30:1293–1299, 2008.

[27] H. He, J. Wang, W. Graco, and S. Hawkins. Application of neural networks to detection of medical fraud. *Expert Systems with Applications*, 13:329–336, 1997.

[28] H. He, W. Graco, and X. Yao. Application of genetic algorithms and k-nearest neighbour method in medical fraud detection. In *Proceedings of the 1998 International Conference on Simulated Evolution and Learning (SEAL)*, pages 74–81, 1998.

[29] N. Immorlica, K. Jain, M. Mahdian, and K. Talwar. Click fraud resistant methods for learning clickthrough rates. In *In Proceedings of the 1st Workshop on Internet and Network Economics*, pages 34–45, 2005.

[30] N. Japkowicz. Supervised versus unsupervised binary-learning by feedforward neural networks. *Machine Learning*, 42:97–122, 2001.

[31] T. Joachims. Making large-scale support vector machine learning practical. In B. Schölkopf, C. J. C. Burges, and A. J. Smola, editors, *Advances in Kernel Methods*, pages 169–184. 1999.

[32] T. Joachims. Estimating the generalization performance of an SVM efficiently. In *Proceedings of the 17th International Conference on Machine Learning (ICML)*, pages 431–438, 2000.

[33] T. Joachims. *Learning to Classify Text Using Support Vector Machines: Methods, Theory and Algorithms*. Kluwer Academic Publishers, 2002.

[34] M. V. Joshi, V. Kumar, and R. C. Agarwal. Evaluating boosting algorithms to classify rare classes: Comparison and improvements. In *Proceedings of the 2001 IEEE International Conference on Data Mining (ICDM)*, pages 257–264, 2001.

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

[36] T. M. Khoshgoftaar, J. Van Hulse, and A. Napolitano. Comparing boosting and bagging techniques with noisy and imbalanced data. *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, 41(3):552–568.

[37] J. Kim, A. Ong, and R. Overill. Design of an artificial immune system as a novel anomaly detector for combating financial fraud in retail sector. In *Proceedings of the 2003 Congress on Evolutionary Computation (CEC) - Volume I*, pages 405–412, 2003.

[38] M. J. Kim and T. S. Kim. A neural classifier with fraud density map for effective credit card fraud detection. In *Proceedings of the 2002 International Conference on Intelligent Data Engineering and Automated Learning (IDEAL)*, pages 378–383, 2002.

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

[40] D. Li, Z. Aung, J. Williams, and A. Sanchez. Efficient authentication scheme for data aggregation in smart grid with fault tolerance and fault diagnosis. In *Proceedings of the 2012 IEEE Power and Energy Society Conference on Innovative Smart Grid Technologies (ISGT)*, pages 1–8, 2012.

[41] D. Li, Z. Aung, J. Williams, and A. Sanchez. P3: Privacy preservation protocol for appliance control application. In *Proceedings of the 3rd IEEE International Conference on Smart Grid Communications (SmartGridComm)*, pages 294–299, 2012.

[42] D. Li, Z. Aung, S. Sampalli, J. Williams, and A. Sanchez. Privacy preservation scheme for multicast communications in smart buildings of the smart grid. *Smart Grid and Renewable Energy*, 2013. in press.

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

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

[45] J. Lin, M. Hwang, and J. Becker. A fuzzy neural network for assessing the risk of fraudulent financial reporting. *Managerial Auditing Journal*, 18:657–665, 2003.

[46] Y. Lin and G. Lee, Y.and Wahba. Support vector machines for classification in nonstandard situations. *Machine Learning*, 46:191–202, 2002.

[47] B. Little, W. Johnston, A. Lovell, R. Rejesus, and S. Steed. Collusion in the US crop insurance program: Applied data mining. In *Proceedings of the 2002 ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 594–598, 2002.

[48] J. Major and D. Riedinger. EFD: A hybrid knowledge/statistical-based system for the detection of fraud. *The Journal of Risk and Insurance*, 69:309–324, 2002.

[49] A. Metwally, D. Agrawal, and A. El Abbadi. Duplicate detection in click streams. In *Proceedings of the 14th ACM International Conference on world Wide Web (WWW)*, pages 12–21, 2005.

[50] A. Metwally, D. Agrawal, A. El Abbadi, and Q. Zheng. On hit inflation techniques and detection in streams of web advertising networks. In *Proceedings of the 27th International Conference on Distributed Computing Systems (ICDCS)*, pages 52–52, 2007.

[51] K. Morik, P. Brockhausen, and T. Joachims. Combining statistical learning with a knowledge-based approach - a case study in intensive care monitoring. In *Proceedings of the 16th International Conference on Machine Learning (ICML)*, pages 268–277, 1999.

[52] B. Neupane, K. S. Perera, Z. Aung, and W. L. Woon. Artificial neural network-based electricity price forecasting for smart grid deployment. In

Proceedings of the 2012 IEEE International Conference on Computer Systems and Industrial Informatics (ICCSII), pages 1–6, 2012.

[53] Panda Security. Virus encyclopedia, 2011. URL http://www.pandasecurity.com/homeusers/security-info/118189/Clickbot.A/. [Online; accessed 10-March-2013].

[54] K. S. Perera, B. Neupane, M. A. Faisal, Z. Aung, and W. L. Woon. A novel approach based on ensemble learning for fraud detection in mobile advertising. In *Proceedings of the 2012 Workshop on Fraud Detection in Mobile Advertising (FDMA)*, 2012.

[55] C. Phua, D. Alahakoon, and V. Lee. Minority report in fraud detection: classification of skewed data. *SIGKDD Explororations Newsletter*, 6:50–59, 2004.

[56] C. Phua, E. Y. Cheu, G. E. Yap, K. Sim, and M. N. Nguyen. Feature engineering for click fraud detection. In *Proceedings of the 2012 Workshop on Fraud Detection in Mobile Advertising (FDMA)*, 2012.

[57] Price Water House Coopers. IAB internet advertising revenue report. URL http://www.iab.net/insights_research/industry_data_and_landscape/adrevenuereport.

[58] J. R. Quinlan. Improved estimates for the accuracy of small disjuncts. *Machine Learning*, 6:93–98, 1991.

[59] M. K. Reiter, V. Anupam, and A. Mayer. Detecting hit shaving in click-through payment schemes. In *Proceedings of the 3rd conference on USENIX Workshop on Electronic Commerce (WOEC) - Volume 3*, page 13, 1998.

[60] S. Shivashankar and P. Manoj. Hierarchical committee machines for fraud detection in mobile advertising. In *Proceedings of the 2012 Workshop on Fraud Detection in Mobile Advertising (FDMA)*, 2012.

[61] L. Sinclair. Click fraud rampant in online ads, says Bing. 2011. URL http://www.theaustralian.com.au/media/click-fraud-rampant-in-online-adssays-bing/story-e6frg996-1226056349034.

[62] B. Stefano and F. Gisella. Insurance fraud evaluation: A fuzzy expert system. In *Proceedings of the 10th IEEE International Conference on the Fuzzy Systems (IFSC) - Volume 3*, pages 1491–1494, 2001.

[63] Y. Sun, M. S. Kamel, A. K. C. Wong, and Y. Wang. Cost-sensitive boosting for classification of imbalanced data. *Pattern Recognition*, 40:3358–3378, 2007.

[64] K. M. Ting. A comparative study of cost-sensitive boosting algorithms. In *Proceedings of the 17th International Conference on Machine Learning (ICML)*, pages 983–990, 2000.

[65] C. J. van Rijsbergen. *Information Retrieval*. Butterworth-Heinemann, 2nd edition, 1979.

[66] C. Walgampaya and M. Kantardzic. Cracking the smart clickbot. In *Proceedings of the 13th IEEE International Symposium on Web Systems Evolution (WSE)*, pages 125–134, 2011.

[67] C. Wei and D. Patel. Hybrid models for click fraud detection in mobile advertising. In *Proceedings of the 2012 Workshop on Fraud Detection in Mobile Advertising (FDMA)*, 2012.

[68] G. M. Weiss and F. Provost. Learning when training data are costly: The effect of class distribution on tree induction. *Journal of Artificial Intelligence Research*, 19:315–354, 2003.

[69] R. Wheeler and S. Aitken. Multiple algorithms for fraud detection. *Knowledge-Based Systems*, 13:93–99, 2000.

[70] G. Williams. Evolutionary hot spots data mining: An architecture for exploring for interesting discoveries. In *Proceedings of the 1999 Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD)*, pages 184–193, 1999.

[71] B. Zadrozny and C. Elkan. Learning and making decisions when costs and probabilities are both unknown. In *Proceedings of the 7th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 204–213, 2001.

[72] L. Zhang and Y. Guan. Detecting click fraud in pay-per-click streams of online advertising networks. In *Proceedings of the 2008 The 28th International Conference on Distributed Computing Systems (ICDCS)*, pages 77–84, 2008.

[73] Z. H. Zhou and X. Y. Liu. Training cost-sensitive neural networks with methods addressing the class imbalance problem. *IEEE Transactions on Knowledge and Data Engineering*, 18:63–77, 2006.

[74] M. Zhu. Recall, precision and average precision. Technical report, University of Waterloo, 2004.