Faculty of Engineering and Information Technology

University of Technology Sydney

Data Mining of Classification for Sybil User Detection

Thesis submitted in partial fulfilment of the requirements for the degree of **Master of Analytics by Research**

By

Anand Arun Chinchore

June 2016

Dedicated

To my late father....

Mr Arun Yashwant Chinchore

CERTIFICATE OF AUTHORSHIP / ORIGINALITY

I certify that the work in this thesis has not previously been submitted for a degree nor has it been submitted as a part of requirements for a degree except as fully acknowledged within the text.

I also certify that the thesis has been written by me. Any help that I have received in my research work and the preparation of the thesis itself has been acknowledged. In addition, I certify that all information sources and literature used are indicated in the thesis.

Signature of Candidate

Acknowledgements

I would like to express my special appreciation and thanks to my supervisor Dr Guandong Xu. You have shown tremendous patience with me as one of my mentors. I would like to thank you for encouraging my research and for allowing me to grow as a researcher in analytics. Your advice on both my research and my career have been priceless.

I would also like to extend my appreciation to my co-supervisor Dr Frank Jiang for providing me with continuous support throughout my Master of Analytics by research and study. Without your professional guidance, persistent help, patience, motivation and immense knowledge, this thesis would not have been possible. Words are not enough.

I thank the management team members of the Advanced Analytics Institute, Mr Colin Wise and Mr Austin Andrade, for their encouragement and guidance towards the completion of my research journey. All of you have been there to support me, when required, over the last two and half years.

I would also like to thank the many other faculties, staff, lecturers, professors and doctors of the University of Technology Sydney, who guided and helped me along the path of my research within UTS.

I thank my workplace manager, and my colleague, Mrs Roula Christodoulides who helped and supported me to balance my time in work and study.

A special thanks to my mother Smt Vasudha Arun Chinchore and my family. Words cannot express how grateful I am to you for all of the sacrifices that you’ve made on my behalf. Your prayers for me have sustained me thus far. I would also like to thank all of my friends who have supported me in writing and given me incentive to strive towards my goal.

Anand Arun Chinchore

June 2016 @ UTS

Table of contents

List of figures

List of Tables

List of publications

Papers published

Chinchore, A, Jiang, F, Xu, G 2015, ‘Intelligent Sybil attack detection on abnormal connectivity behaviour in mobile social networks’, in the proceedings of the Springer International Publishing 10th International Conference, KMO (2015), Maribor, Slovenia, 24-28 August, vol. 224, pp. 602-617.

Abstract

Data analytics and Big Data application research, along with new structures in complex data, are reveling the secrets of their own complexity and patterns with valuable and critical, but challenging, issues through newly designed tools, techniques and models in data science technology. A common example concerns the interconnectivity of social network users on mobiles, involving content and information sharing through mobile social networks.

There have been a large number of studies on mobile networks. Many focus on a variety of secured applications that attempt to exploit social connections, impersonate users or attack social groups. Such applications are often created with the intention of collecting confidential information, laundering money, blackmail or to perform other criminal activities.

Existing methods for identifying such activity, such as distributed systems, social graph-based sybil detection, behaviour classification, and local ranking systems that estimate the trust level between users, rely on the dependencies between random nodes of connection on mobile social networks. These models aim to detect suspicious connections and have the advantage of learning the relationships between nodes and data. However, their detection patterns tend to impose the behavioural patterns typically associated with community-based and external networks.

In data mining, the graph-based and classification models used for pattern collection can accurately predict patterns in data in targeted categories. Decision trees, commonly used for classification, are trees in which each branch represents a choice between a number of alternatives, and each leaf represents a classification, or decision. For example, a decision tree may help an institution decide whether a node in a dataset is suspicious, or considered to be sybil, if a

decision tree can be induced from a set of data about its instances and the - classifications of those instances. It could also provide the flexibility to demonstrate data distribution. Thus, researchers have tried to combine different techniques and methods into network-based models to detect various patterns generated by sybil nodes within a network.

The purpose of this thesis is to abridge existing classification and regression techniques to identify sybil nodes, and the correlation of those nodes with time, to address these research limitations.

Classification and regression techniques predict behaviour based on continuous or categorical responses. For example if the predicted response is continuous, then it is called a regression tree. If the response is categorical, it is called a classification tree. At each node of the tree, the value of one the connected input nodes is checked and a binary answer − yes or no − determines whether one continues to the left or right sub-branch. When a leaf is reached, a prediction follows from a series of entropy calculations and graphing techniques.

This thesis introduces a novel classification model for sybil detection in mobile social behaviour that identifies dependencies using connection duration and other attributes. Roger Quinlan's C4.5 algorithm, its resulting decision tree and a random forest simplify the step-by-step identification process, while maintaining its merits. Partial correlations between nodes are simplified using Rattle programming, and the dataset is divided into majority nodes to assist processing.

This research also includes a behavioural survey of the nodes and an extended analysis using a classification system for sybil detection, with a particular focus on sybil attacks in mobile social network environments. Each sybil node is tracked and identified based on the frequency and duration of its connections with other nodes.

An outline of how the classified model identifies behaviour is also included, along with an explanation of the flow of the decision tree and the C4.5 algorithm process, which press-gangs identified sybil nodes based on the results of entropy calculations and information gain. The calculated entropy for each node connection across the all datasets informs the information gain. The maxGain calculations for individual node bring the final stage of draw decision tree and helped to predict the sybil nodes, compare and justify the sybil attackers .

These processes and new models applied to sybil detection provide insight into the behaviour of connections, through deep analytics and entropy gain. The evidence gleaned from this research brings significant knowledge to data analytics and data science in the identification of threats on mobile social networks.

Chapter 1
Introduction

1.1 Background

There have been a large number of studies on mobile networks. Many focus on a variety of secured network applications that exploit social connections, present false identities or attack social groups. These applications are created with the intention of collecting confidential information, laundering money, blackmail or to perpetrate criminal activity.

Over the last decade, an influx of complicated and varyingly structured data has flooded the internet. As a result, common users of mobile apps and technology are quickly occupying the personalised service market. Uploading data, social networking, banking systems, and the storage and production of text, images, animations and confidential information on mobile technology is easier than that was on computers and laptops. Additionally, data distribution through mobile and social networks is easier than via email. Mobile technology's ability to save time and its high appeal to human psychology and behaviour have been recognised (Huiqi, Z. et al, 2011). Additionally, data compression has become easier due to smart phone technology, and is a free and quick alternative to computer software. Free apps and inbuilt mobile technology have made human social, financial, business and cultural and life easier than before.

The improved performance this mobile techngology has brought to the world derives from a combination of research sectors involved in highly innovative study across science and deep analytics. Connecting to each other through networks, inviting others to your network, exchanging confidential information or gossiping on chat, and even creating your own chat applications is as easy as a click of the button. Such processes have been made possible by researchers and developers with the aim of capturing every thought and human action required to meet the needs of daily life.

As much as research has contributed to innovation, it has also contributed to the security of networks, applications and devices. While individuals may like to have their thoughts and actions computerised, they also demand strong protection of their data, personal information and life.

1.1.1 Mobile social networking

Mobile social networking allows individuals with similar interests to connect with one another through their mobile phones and tablets. Much like web-based social networking, mobile social networking takes the form of virtual communities. Social networking websites, such as Facebook, create mobile apps to give their users instant and real-time access to these networks from their mobile devices, while users create additional applications, such as Foursquare, Instagram and Path, to increase mobile functionality for their community. The line between mobile devices and the web is becoming more and more indistinct, as mobile apps use existing social networks to create native communities and promote discovery, while web-based social networks taking advantage of mobile features and accessibility.

The advent of fully mobile access to the internet has brought distinct changes to web-based social networks. Many are being extended, for mobile access, through mobile browsers and smart phone apps, as native mobile social networks. A dedicated focus on mobile use, like chatting apps, GPS, and increased reality, requiring mobile devices and technology on virtual world (Cortimiglia, M. et al, 2011). Mobile- and web-based social networking systems often work symbiotically to spread content, increase accessibility and connect users from wherever they are.

This brings safety issues, including trust, privacy and security in mobile social networks to the fore. Chang W. et al (2013) points out that mobile users contacting each other in mobile environments also raises concerns for protection against different types of failure, damage, error, accidents, harm and other nondesirable events. However, a lack of protective infrastructure in these networks has turned them in to convenient targets for peril. This is the main impulse behind many mobile social network's disparate and intricate safety concerns, and their desire to embrace divergent safety challenges(Chang W. et al, 2013). It is also the environment that brings the researchers, scientists and investigators into the picture.

1.1.2 Social media and data mining

A social networking service (SNS) is an online service or web sites that focuses on facilitating the building of social networks or social relations among people who wish to share same interests, activities or connections. Companies often create customer profiles from generated data through social media that contain customer demographics and online behaviour. Data miners can use those data collections to find patterns and information, then design new techniques and methods to improve social networking sites and their protection. Mobile communities are generally united by shared interests or goals. They interact by means of location-independent information technology, and this includes mobile access to existing community infrastructures. Various models, based on these data collections, have been adopted by networking sites, and while most offer a unique feature or special function that other sites do not share, the main functions of the site are exactly the same as other services.

1.1.3 The threats in social networking sites

SN users notice that they are giving too much social support to other SNS friends and unknown people in between connections. The increasing number of messages and social relationships rooted in an SNS also increases the amount of social information demanding a reaction from SNS users. The more information that is shared on SNSs, the more potential there is for an individual's privacy to be violated. The threat multiplies when information shared on one's own profile is then re-shared by others and the owner loses control over it.

1.2 History of the term sybil

The term sybil was coined after the “Sybil” a famous book and subsequent TV mini-series of the same name. They tell the true story of patient with multiple personality disorder, which is now known as dissociative identity disorder (DID) in medical terminology. DID is a rare dissociative disorder in which two or more personalities with distinct memories and behaviour patterns apparently exist in one individual. The manifestation of multiple personalities is hard to understand even for very proficient specialists. Its development, analysis, diagnosis and examination continue to be debated among psychological health professionals. Some experts believe it is: an *offshoot* phenomenon of another psychiatric problem, such as borderline personality disorder; the product of profound difficulties in coping abilities; or stresses related to how people form trusting emotional relationships with others. It is characterised by people who have two or more distinct or split identities that continuously have power over the person and their behaviour.

1.2.1 Sybil users in computer terminology

Technology makes the work of attackers much simpler, and facilitates the theft of information from mobile social networking groups in a fraction of a second without the need for any special hacking software. Sybil users are those who create a false identity for the purposes of data distribution on the internet. Such users are called a sybil user on mobile social networks too. They pretend to be an honest entity to collect information from a legitimate entity, with the intention of blackmail, money laundering, hacking, or to perpetrate other criminal or offensive activities. Data may be taken for their own use or to distribute amongst other sybil users.

1.2.2 Honest users and sybil users

Honest nodes, or users, are authentic nodes connected to other honest nodes. Sybil nodes, or users, are those using a false identity or profile designed to steal information. False nodes can also be connected to other false nodes for the purposes of exchanging data collected between sybil users, and this behaviour can be tracked. Research tools, like connectivity analysis, can help to identify IP address spoofing, the total number of social contacts, duplicate IPs, the time of connection, the number of connections, and all relevant attributes in the hunt for sybil activity.

Sybil users are those who create a false identity on mobile social network for the intention of data distribution. A single user or group of sybil users may intend to harm the social media platform, a particular organisation, or an individual. The intention behind their activity can range from: reducing the revenue of an organisation; stealing customer data to sell to a competitor; blocking or hampering a social media provider and their services to decrease user involvement or the registration of new users; to misuse an individual's data to a achieve personal objective; and many others.

1.2.3 Sybil user communities and group forming

Two or more sybil users may collude to form a group or community of sybils, whose purpose is to generate high levels of destruction or harm to individual or organisations data. These attacks can be executed by sybil users pretending to be honest users in an honest community. One or more sybil users may involve themselves in an honest community, pretend to be an honest user, and then pass the information to the other sybil users, or the sybil community or to the honest community. The variety of profiles this creates can make it hard to distinguish the sybil users from the honest ones amongst the honest region as they were just the same as honest characters. Furthermore, as Zhang K. et al (2013) notes, "Sybil users generate connections with other sybil users from honest regions to sybil regions and these can be difficult to identify, as sybil entities could be using a changing IP location, or a false IP addresses by special tools and software."

1.2.4 Sybil users, sybil behaviours and sybil attacks

Little research has been published on identifying the abnormal behaviour of sybil users, yet the intention to cause harm is common in human psychology. This kind of behaviour may result from mind-disturbing events in one's past, as in the case of the real life Sybil, or may be just for fun. In real life, this type of abnormal behaviour can be discerned. For example, when a couple breaks up or when a person launders money. Sybil users perform similar actions in the virtual world.

Abnormal behaviour can be observed as the following scenarios: the breakup of a couple, resisting attack, money laundering, stealing individual information for personal use, performing a terrorist attack, collecting data from multiple users to sell to other organisation to generate revenue, reducing the revenue of an organisation, causing harm to an individual's life or an organisation for personal reasons, or as a result of many human psychological barriers.

Such behaviour ranges from low intensity, such as stealing data from an individual, to very high intensity, such as a terrorist attack in the real world using social media as its communication channel. Similarly, some behaviour can be seen as childish teasing, or born of ignorance or poor education. These users are mostly aged between 14 and 22 years old, and perform actions either due to lack of knowledge about the law, or are trained for specific acts by other professional sybil attackers.

1.3 Classification and regression for data mining

A huge amount of data is being collected and stored in databases across the world and its space stations, and this trend continues to increase year upon year. So much valuable knowledge is hidden in these database, it is practically impossible to mine them without an automatic extraction method. Over the years, many algorithms, called nuggets, have been created to extract this knowledge using various methodologies such as classification, association rules, clustering, and many more.

Classification consists of predicting a certain outcome based on a given set of inputs. Typically, an algorithm processes a training set, containing a set of attributes and the respective outcome, to discover the relationships between the attributes that make the outcome possible. The algorithm is then given an unseen dataset, called the prediction set, which contains a similar set of attributes without the outcome. The algorithm analyses that input and attempts to produce a prediction.

Classification models help to predict categorical class labels, which may be discrete or nominal. Constructed models classify data based on a training set and use the resulting class label values as attributes with which to classify new data.

Decision trees are a very popular method of classification for supervised learning with nominal classes that have dependent labels. Using decision trees to predict nominal class behaviour given a simple 'yes' or 'no' is straightforward. However in some cases, nominal classes can specify more than two options, and hence which kind of entities responded to which options can be classified and/or predicted.

The decision tree for the classification technique outlined in this thesis was built by first searching for users identified as suspicious. Users were then split into the mostly evenly divided groups given a set of observations and their features. Mixed data of User 1 and User 2 connections with their starting time, ending time including other columns in dataset separated and categorised on User1 and their respectives. However, this split may not always be symmetrical or even, given the number of user connections will change from node to node, as will the connection time between two nodes.

Decision trees work very differently than naïve Bayes. When attempting to predict a sybil user using a decision tree, all the nodes and their possible connections need to be observed. For example, the Infocom06 dataset records 98 users, connecting to over 4000 users multiple times, which generates more than 200,000 connections. The set also includes the length of time the nodes were connected, along with other attributes.

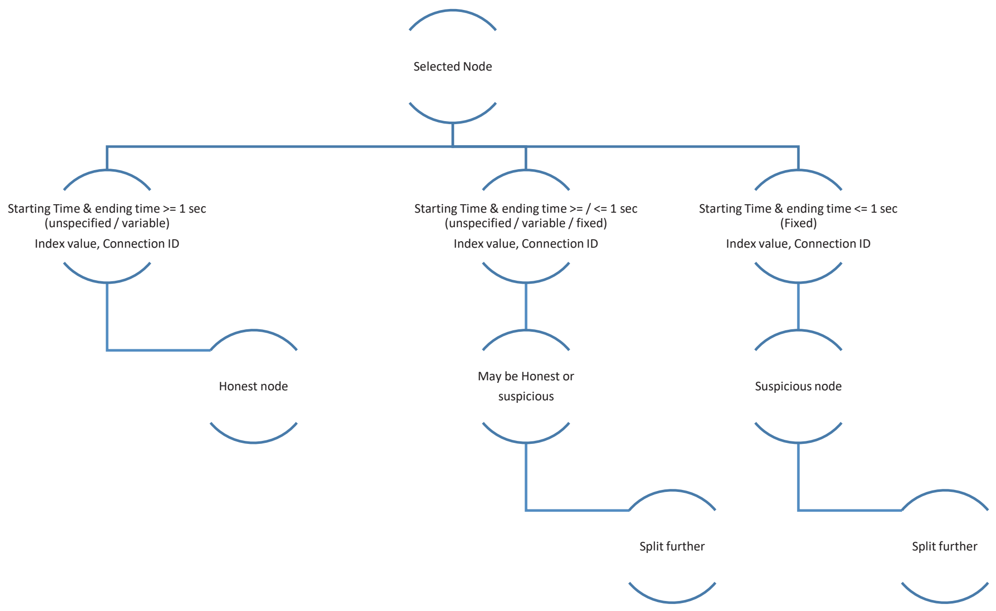

Selected Node
Starting Time & ending time >= 1 sec (unspecified / variable)
Index value, Connection ID
Starting Time & ending time >= / <= 1 sec (unspecified / variable / fixed)
Index value, Connection ID
Starting Time & ending time <= 1 sec (Fixed)
Index value, Connection ID
Honest node
May be Honest or suspicious
Suspicious node
Split further
Split further

Figure 1.1 Decision tree map node distribution Infocom06 dataset

These decisions, determine the trees that are selected for further random forest processing. The key to generating a decision tree for each node is to determine what type of connections the node has made and why it made those connections.

The number of connections a given node has made could be: specific (ie regular/honest); limited (ie only few connections); or large (ie connect to most of available or more than selected average). Those three subsets will contain even further subsets of information.

If subset is pure, then it is honest. That means if the node had specific connections, and connected over an unspecified time, the process will stop. Otherwise it will continue to investigate the behaviour and try to further split the data – a sort of variation on a divide and conquer algorithm. Further, if the selected node is categorised as suspicious or considerably suspicious, the node will be divided into even more branches. True values will split the tree further, false values will deem the node honest.

1.3.1 C4.5 algorithm

The C4.5 algorithm was developed by Ross Quinlan to generate decision trees. C4.5 is an extension of Quinlan's earlier ID3 algorithm. The decision trees generated by C4.5 can be used for classification, and for this reason, C4.5 is often referred to as a statistical classifier.

The generalised algorithm for building decision trees is:

- Check for basic cases
- For each attribute *a*
- Find the normalised information gain ratio from splitting *a*
- Let *a_best* be the attribute with the highest normalised information gain
- Create a decision node that splits *a_best*
- Recur on the sublists obtained by splitting *a_best*, and add those nodes as children of the node.

1.3.2 Entropy

Calculating entropy is a way to measure the uncertainty of a class in a subset.
Entropy is defined as:
H(S) = -p(+) log2 p(+) – p(-) log2 p(-) bits,
where

- *S* is a subset of the training example
- *p(+) / p(-)* ... is a percentage (%) of the positive / negative examples in S

Interpretation is needed in this context. Assuming item X belongs to S, how many bits are needed to tell if X is positive or negative?
If the subset has, for example, 10 positive and 10 negative samples, then 1 bit is spent. In cases less certain than the pure set, but more certain than a coin flip, entropy gives a number between 0 and 1 which represents a fractional number of bits – except a fractional number of bits is little strange.
Hence, if the impure subset = 1 bit:
H(S) = - (number of purity / total number) log2 (number of purity / total number) - (number of impurity / total number) log2 (number of impurity / total number) = 1 bit

If the pure subset = 0 bits:

H(S) = - (number of purity / total number) log2 (number of purity / total number) - (number of impurity / total number) log2 (number of impurity / total number) = 0 bits

This is how entropy tells us how pure and impure is one set and subset.

1.3.3 Information gain

Information gain, or mutual information, is a synonym for Kullback–Leibler divergence in information theory and machine learning. In the context of decision trees, the term is sometimes used synonymously with mutual information, which is the expectation value of the Kullback–Leibler divergence of a conditional probability distribution.

The information gain about a random variable X obtained from an observation that a random variables which is chosen to build each decision tree. This concept is used to define the chosen structure of attributes to observe that most rapidly narrow down the state of X. Such sequences depend on analysis of the previous attributes of the decision tree.

$$Gain\ (S, A) = H(S) - \sum_{V \in value(A)} \frac{|Sv|}{|S|}\ H(Sv)$$

V is a possible or particular value of A

S is a set of *fi* (example) {X}

Sv is a subset where X_A= V, which is a total of the all the connections.

This is taken into account when adding the entropy value, by placing a weight on each entropy. We put a weight on each subset and the weight is the size of that subset divided by the overall number of *fi* that we have at this split node. If we get big pure subset then it’s good or if we get small or big impure subset then its bad.

In summary, the average purity is weighted by the size of the set’s average purity after the split on attribute A. Looking at the difference in entropy before and after the split is the determining point at which interpret. We have to know how much we are certain before split and how much more certain we are after the split. So based on this decision, the node is considered as honest or which going to be consider as, suspicious or sybil.

1.3.4 Random forest

Random forest is an ensemble-style, learning-based classification and regression technique, and is commonly used for predictive modelling. A single decision tree is built while decision tree but in a random forest, number of decision trees are to build at once and create a forest. A vote from each of the decision trees decides the final class of a object. This is called an ensemble process, or a collective process, where decisions are based on the number of pure and impure subsets. Any decision trees built and used as part of a random forest algorithm become a part of the forest.

Data frames have two dimensions, observations (rows) and variables (columns). To build a decision tree, samples of a data frame are selected is a replacement along with the selection of a subset of variables for each of the decision trees. Both the sampling of the data frame and the selection of the subset of variables are done randomly.

There are two key advantages for using random forests in analytics. The chances of over-fitting are reduced, and they provide higher model performance and accuracy when used with classification and regression problem identification. Random forests use Gini index-based impurity measures to build decision trees. Gini indexes are also used to build classification and regression trees (CART).

1.4 Research aims

This research aims to identify sybil users or groups of sybil users in a complicated environment. The study evaluates a dataset to find connections among its users that distinguish suspicious activity performed by false identities to improve social media, and reduce its threats to individuals and organisations. Many information sources within the domain were reviewed to enrich the quality of this research to ensure it plays a significant role in social media analytics.

The research proposed in this paper resolves the following issues:

1) Does abnormal behaviour between nodes on mobile social networks distinguish connections between honest and sybil nodes?
2) Do irregular times of connection between nodes play a vital role in identifying this abnormal behaviour?
3) Does the connectivity intensity or frequency of connections help to identify sybil users?

Taking dataset availability and the records within it into consideration, the above research problems can be evaluated using the Infocom06 dataset. This dataset contains User 1 connection (total 98 users) and connection to User 2 (over 4000 users) connected multiple times to user one. Each connection records a start time, an end time and its index value.

Final outcomes are explained in the form of algorithm, graphs, charts and a table that displays information about each node including assumptions, sybil category, reasons, and each node connected other nodes.

The aim is to developed a graph-based system that shows the connectivity intensity - the number of times honest nodes connect to sybil nodes and sybil nodes connect to other sybil nodes. This first stage of research was published as paper at the KMO2015 conference.

In the second stage of research, the dataset was evaluated using classification and regression techniques, decision trees, a random forest, the C4.5 algorithm, entropy generation, and an information gain model.

This thesis explains how sybil attacks can be traced with the help of a graph-based system and how classification and regressions techniques can be used to predict sybil behaviour. To identify sybil users, amongst honest users, a cross-checking method examine the number of times a connection occurs with a single user or multiple users. By way of practical explanation, first, suspicious activity by false identities was determined by at least one, or a regular, or a continuous connection with single or multiple users for a longer than usual time which helped to identify sybil users. Second, an irregular time of connection between two nodes creates suspicious activity between nodes. To identify this, algorithm to get into use and improve social media and reduce its threats for others, this research presented how graph-based system helps to identify the connectivity intensity or the number of times of connections between honest node to identify sybil node. Research also predict behaviour of sybil nodes to other sybil nodes connectivity and explains how they made connections.

Significance 1: Identification of false identities using a graph-based technique

This research produces a set of graphs, from a typical dataset, that can be used to analyse, and ultimately identify, sybil users with marginally better performance and accuracy than existing methods. The graphs provide a range of perspectives on the number, frequency and duration of connections between nodes that, when combined with classification techniques, help to evaluate internode relationships and distinguish sybil behaviour.

Significance 2: Ability to track the behaviour of sybil nodes

Using behavioural techniques, the research outcomes demonstrate that sybil users can be identified based on the duration of connectivity between each user – from the start to end time of each connection, and the number of connections with a single user. These results complement Significance 1 and and help achieve better success in tracing sybil users in the dataset.

1.5 Research contributions

- A proposed algorithm to identify sybil activity using a graph-based system, with generated graphs of connections and attributes (Chapter 3);
- Behaviour prediction from graphs and observed node connections; and design of algorithmic statistical software and generated graphs to identify connectivity behaviour (Chapter 3);
- A proposed algorithm base on classification and regression techniques for each node showing development of a processing algorithm to build a number of decision trees (Chapter 4);
- Analysis of the decision tree model, the C4.5 algorithm and pass entropy model to generate entropy for each leaf node based on attributes (Chapter 4); which leads to analyses and calculate of the entropy and gain information for each node (Chapter 4);
- Processing of all trees and construction of a random forest for selected variables with plots based on model accuracy and Gini values (Chapter 4);

- Analysis of maximum information gain based on each node and entropy (Chapter 4); this proposed a contribution to research help to minimize the unwanted data.
- Combined maxGain and reprocessing of the decision tree (Chapter 4); and table for
- Analysis of entropy and maximum gain from the newly built tree and sybil user prediction based on designed models and algorithms (Chapter 4).

1.6 Thesis structure

The thesis structure as follows:

Chapter 2 provides a literature review of sybil detection in mobile social networks, including the models and techniques used. The review outlines the variety of methods and techniques researchers have used to identify sybil attackers in data mining and Big Data models. Graph-based models, classification and regression analysis techniques for decision trees and prediction, and a range of mobile network research is reviewed for methods of identifying sybil connections and patterns.

Chapter 3 presents the designed algorithm and presents an analysis of the graphs used or produced in the research. These graphs show the connectivity of each user. By comparison, the experimental results demonstrate the connectivity of sybil nodes. A paper based on this chapter was published in the proceedings of KMO2015.

Chapter 4 demonstrates the classification and regression techniques used to build the decision trees based on the designed algorithm and pruning, along with further decision trees used to calculate the entropy of each node, and the maxGain in common user-node connections. The importance of using entropy and the gain model is assessed by comparing the node connections based on the time of connectivity and other related attributes. The class of target, or response variable, is factor, so a classification number of decision trees using random forest is built which is later used to plot error rates across the decision trees.

Variable importance plots are also a useful processing tool, and are plotted using the **varImpPlot** function. The **confusionMatrix** function, from the caret package in R, is used to create a confusion matrix based on actual response variables and the predicted value. Additionally, the response to the validation sample is predicted and the model's accuracy with the sample is calculated.

Chapter 5 concludes the thesis and outlines the scope for future work.

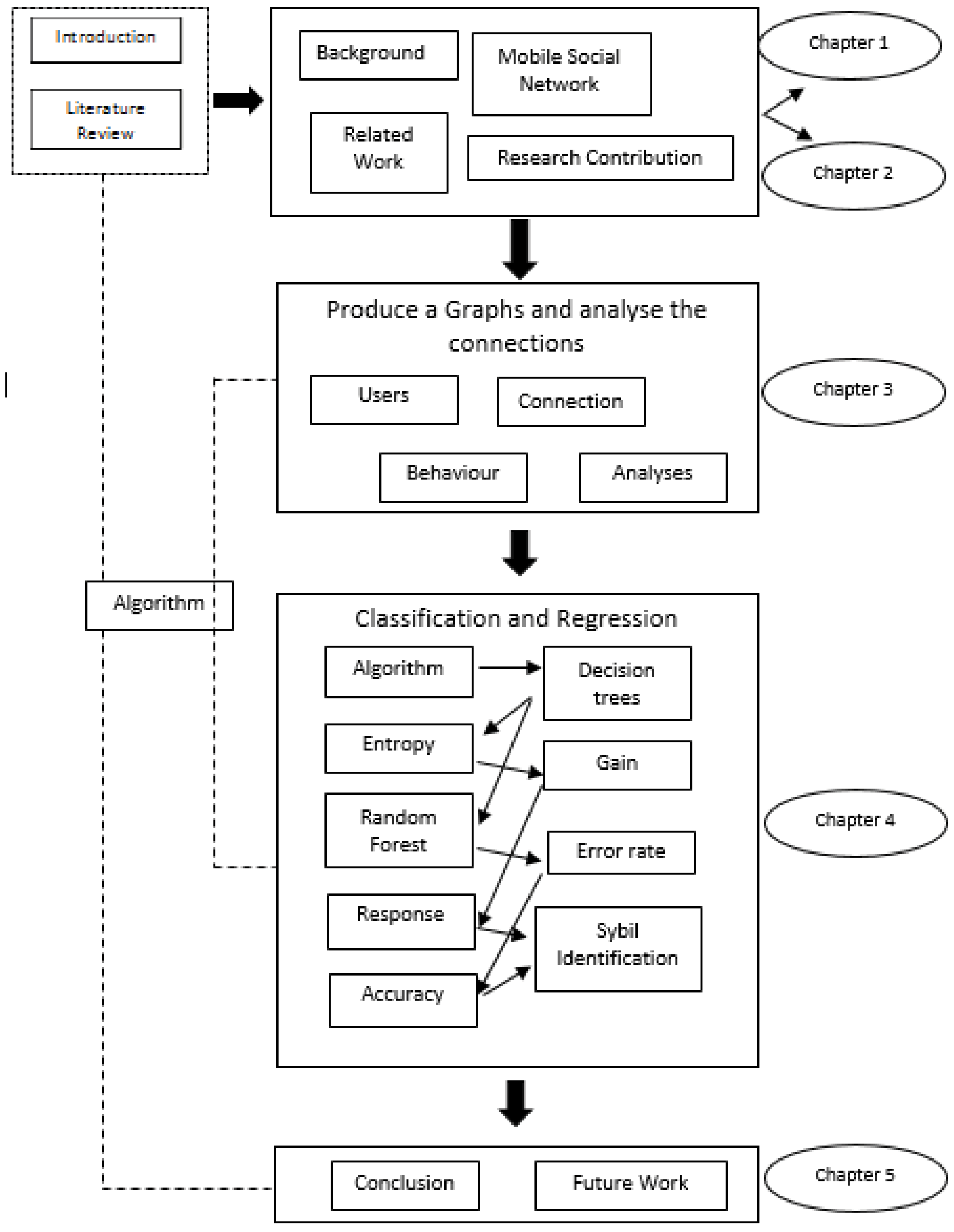


Figure 1.2: Work flow in this thesis

Chapter 2
Literature review and foundation

This chapter reviews work related to sybil detection in mobile social networks. It explores various methods, algorithms, charts, and the types of sybil attacks in sybil detection and behaviour tracking systems. Each article raised important points for consideration in this thesis, and helped to evaluate the research outcomes from the proposed graph-based model and classification and regression techniques. Most of the past published paper considered and studied for this research as there are lot to take and considered while experimented. Even though this thesis presents a novel classification model for sybil detection, the models, techniques, mathematical equations, procedures and concepts in this literature review inspired and informed many of the ideas that were ultimately adopted.

2.1 Directed social networks

Pengfei Liu and his researcher team, 2014, in their study on potential defenses against sybil attacks in directed social networks, proposed a set of measures for the quality of network partitions, and presented an algorithm based on sets of measures and iterative optimisation to detect sybil regions. Modularity was treated as a special case.

Their research paper explains, the researcher tries solving the problem of defence against sybil attacks in directed social network. The presented algorithm based on the set of measures and iterative optimisation to detect the sybil region. The algorithm was evaluated using a subset of real-world social topology and was confirmed and demonstrated to be efficient for solving the problem. Comparisons to Sybil defender confirm their proposed algorithm is superior for sybil region detection in directed social networks.

Their research goes on to explain that social ties and social relationships in microblogging services are very different from those in undirected social networks. “The former can be modeled as directed graphs, since one user can follow another without his or her permission – which, when taken a step further, can be modeled as undirected graphs, one has to be confirmed to become others friends”. The paper describes the problem of defense against sybil attack in directed social network arguing the attackers pattern adopted by sybil users by infinite microblogging services. Further, research proposed some set of defined measures which evaluate network partitions with tenable atom measure function for various applications including further proposed algorithm for detecting sybil region in directed social network evaluate and compare with reference scheme.

The algorithm was evaluated with a subset of Sina Weibo and confirmed to be efficient for the problem, suffering only a 0.017% false positive rate and a 0 false negative rate.

The research provides a small insight into how sybil patterns could be observed in mobile social networks, considering the social ties and relationships between honest nodes.

2.2 Defending against sybil attacks in Mobile Social Network

Yan Sun, Lihua Yin, and Wenamao Liu, 2014, discussed defending sybil attacks in specific types of MSNs based on the past focus researchers. They proposed a security mechanism to detect and eliminate sybil nodes. Their approach measures the distance client-side and eliminates the sybil node server-side. They also demonstrates the solution is correct and analyse its energy costs. Many socially-aware research algorithms have been proposed to address routing problems in MSN, however their study particularly focussed on how to detect and eliminate these virtual nodes to ensure routing security while routing forward.

Their first theorem states, “A malicious node cannot forge a sybil node that has a reasonable location, if it does not know the locations information of its neighbour nodes, but knows the distance information to its neighbour nodes”. The second theorem explains, “If the randomly selected vertices are real nodes, then there is no sybil node in the produced set of routing forward nodes using their Algorithm 1.” While their third theorem specifies, “If there is at least one sybil node in the randomly selected vertices, then the amount of the produced set of routing forward nodes is less than the one of the produced set by all real nodes.”

These theorems and the accompanying proof describe a complete procedure for how to defend against sybil attacks in MSN. Their second algorithm also includes a proposed solution to reduce communications costs by measuring the distance client-side.

2.3 Attack sybil in MSN

More and more evidence shows that some real social networks are not fast-mixing, especially when strong trust relationships are considered. Moreover, the accuracy of all existing solutions is related to the number of attack edges that the adversary can build. This research, proposed by W. Chang and his research team, 2013, explains a local ranking system for estimating the trust level between users. Their scheme has three unique features. 1) Systems is are based on both trustful and distrustful relationships. 2) Instead of storing an entire social graph, users carry limited information related only to themselves. 3) system weakens the impact of attack edges by removing several suspicious edges with high centrality. The effectiveness of their system was validated through comprehensive experiments.

Their design uses a distributed sybil defense algorithm, based on signed networks. To reduce the impact of attack edges, they proposed a gateway breaking algorithm that ‘cuts off’ several high centrality edges from the social graph. The connectivity between honest nodes bears much less of an impact than that between sybil and honest nodes. They proposed in their research “algorithm potentially increases the accuracy of any social network-based sybil defense algorithm and proposed scheme was also based on the unique link structures of sybil nodes, but unlike previous works, they have used more realistic model that the honest users present single or multiple communities".

Their proposed system consists of two parts – a remote server and users. The server is responsible for storing and periodically pruning the signed network graph it creates, as well as assigning randomly sampled social profiles to users for computing the trust-level between users.

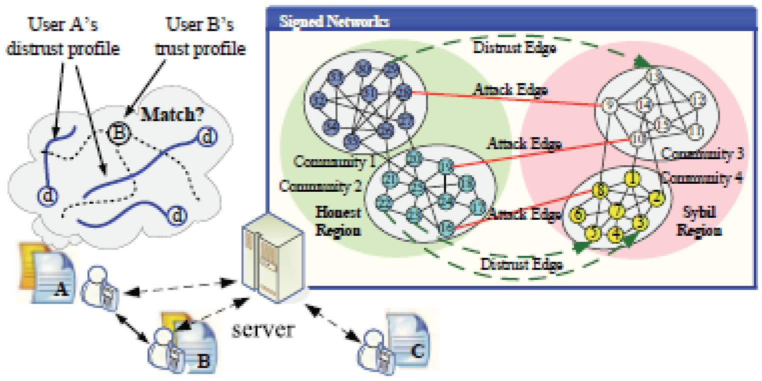


Fig. 2.1 System model of research as explained in Chang W. et al's 2013 research.

In the system flow, each user locally stores two randomly sampled profiles - a trust and a distrust profile. Whenever two strangers encounter each other, and want to establish a cooperative service, they exchange those profiles together with a signature and a timestamp. Both a trust and a distrust score is computed locally to determine whether the other user is a sybil. A special pruning algorithm is kept running on the server to increase accuracy.

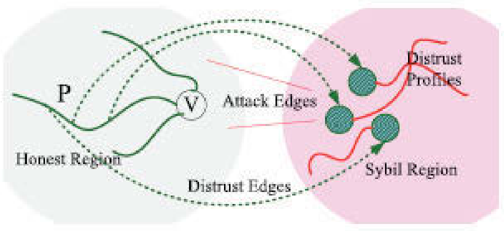


Fig.: 2.2 Generation of a social distrust profile, as explained in Chang W. et al's 2013 research.

Their claimed in proposed algorithm solves the problem of high false positives in existing sybil defense algorithms, and their gateway-breaking algorithm not only increases accuracy, but can be used by any social network in sybil defense. Additionally, extensive simulations have proven the strong performance of this research method and its algorithms.

Jiang J. et al, 2012, researched the detection and validation of sybil groups in online social networks, by considering how sybil users propagate spam or unfairly increase the influence of a target user. However, typically, single users do not harm the system at a high level. Yet, their research does represent the first attempt to identify and validate sybil groups in RenRen online social networks. It identifies over 2600 groups, amongst 985,000 users, by analysing the action time similarity of users within the groups. Their research claims that, "Sybil users alone do not harm the system. What is really dangerous is that multiple sybil users collude together and form a sybil group. The attacker controls the sybil group to attack the system seriously."

Their research designed an automatic validation mechanism for sybil groups, by analysing the action time similarity of users in a group and then further applied those validation methods to sybil groups. Observing that users in sybil groups show an extremely high similarity of action time than that of users in normal groups, they claim that sybil users are simultaneously controlled by the same attacker. Overall, 2440 sybil groups (91.9%) and 985,797 sybil users (99.6%) were successfully validated.

This validation method provided valuable knowledge in developing classification techniques. It guided the selection of attributes for decision trees, and the connectivity, based on time stamps, number of connections and duration of the connection, helped trace links and distinguish sybil users from honest users.

2.4 Sybil defenders

Distributed systems without trusted identities are particularly vulnerable to sybil attacks. Wei W. et al's 2013 paper presented a sybil defence mechanism that leverages network topologies in the defense of sybil attacks in social networks. Based on performing a limited number of random walks within a social group, SybilDefender is an efficient and scalable solution for large social networks. Their research experiments on two 3-million-node, real-world social topologies showed that Sybil defender outperforms the state-of-the-art by one to two orders of magnitude in both accuracy and running time. It can effectively identify sybil nodes, and detect a sybil community around a node, even when the number of sybil nodes introduced by each attack edge was close to the theoretically detectable lower bound. Additionally, they proposed two approaches for limiting the number of attack edges.

The team proposed that sybil attacks can be mitigated by assuming the existence of a trusted authority that requires users to provide some credentials, like a social security number, or a payment. Such an approach would rate-limit the introduction of false identities, but may also deter honest users from joining the network.

Their research elaborates that two users share a link if a relationship is established between them. Each identity is represented as a node in the social graph. To prevent an adversary from creating many sybil identities, all previous sybil defense schemes must be built upon the assumption that the number of links between the sybil nodes and the honest nodes, also known as attack edges, is limited. As a result, although an adversary can create many sybil nodes and link them in an arbitrary way, there will be a small cut between the honest region and the sybil region. The small cut consists of all the attack edges and its removal disconnects the sybil nodes from the rest of the graph, which was leveraged by previous schemes to identify the sybil nodes. The solution to this problem was non-trivial, because finding small cuts in a graph was an NP hard problem. To limit the number of attack edges, previous schemes assumed that all the relationships in social networks are trusted and they reflected the trust relationships among those users in the real world, and thus an adversary cannot establish many relationships with honest users. However, in other research studies it has been shown that this assumption does not hold in some real-world social networks.

To address the weaknesses of previous work, research paper proposed by Wei W. and team, Sybil defender, a centralised sybil defense mechanism. It consists of a sybil identification algorithm to identify the sybil nodes, a sybil community detection algorithm to detect the sybil community surrounding a sybil node, and two approaches to limiting the number of attack edges in online social networks. Their scheme was based on the observation that a sybil node must go through a small cut in the social graph to reach the honest region. An honest node, on the contrary, was not restricted.

The research evaluation showed that Sybil defender correctly identify identified the sybil nodes even when the number of sybil nodes introduced by each attack edge approached the theoretically detectable lower bound, and it effectively detected the sybil community surrounding a sybil node with different sizes and structures.

The survey results of their Facebook application show that their previous assumption: ‘all the relationships in social networks are trusted’ does not apply to online social networks, and it is feasible to limit the number of attack edges in online social networks using relationship ratings.

The study also proved that the sybil identification algorithm and the community detection algorithm are efficient and scalable to large social networks. Sybil defender was evaluated using two large-scale social network samples from Orkut and Facebook. The results showed that the performance of the sybil identification algorithm approached the theoretical bound, and outperformed sybil limit, the state-of-the-art sybil defense mechanism for large social networks, by one to two orders of magnitude in both accuracy and running time. Their presented approach sybil community detection algorithm has effectively detected the sybil community around a sybil node with short running time.

The proposed research also had proposed two practical techniques to limit the number of attack edges in online social networks, and has developed a Facebook application to demonstrate the feasibility of one of the techniques. The survey “results of the Facebook application showed that the assumption made by previous work that all the relationships in social networks are trusted does not hold in online social networks, and it is feasible to limit the number of attack edges in online social networks by relationship rating” (Wei W. et. al., 2013).

2.5 Sybil limit

Haifeng Yu et al, 2010 claim that: “without a trusted central authority that can tie identities to real human beings, defending against sybil attacks is quite challenging. Among the small number of decentralised approaches, our recent sybil guard protocol leverages a key insight on social networks to bind the number of sybil nodes to be accepted.”

This paper presented “a novel Sybil limit protocol that leverages the same insight as Sybil guard, but offers dramatically improved and near-optimal guarantees. The number of sybil nodes accepted is reduced by a factor $\theta(\sqrt{n})$, or around 200 times in experiments for a million-node system. They further proved that, Sybil limit’s guarantee was at most a log n factor away from optimal when considering approaches based on fast-mixing social networks. Finally, based on three large-scale real-world social networks, they provided the first evidence that real-world social networks are indeed fast-mixing”.

Furthermore, their research states that “sybil attacks can be thwarted by a trusted central authority if the authority can tie identities to actual human beings, but implementing such a capability can be difficult or impossible, especially given the privacy concerns of the users”.

Recently their proposed research the sybil guard approach explains, sybil guard, is a new protocol for defending against sybil attacks without relying on a trusted central authority. “Sybil guard leverages a key insight regarding social networks. In a social network, the vertices (nodes) are identities in the distributed system and the (undirected) edges correspond to human-established trust relations in the real world. The edges connecting the honest region (i.e., the region containing all the honest nodes) and the sybil region are called attack edges”. It also claims, “first, although the end guarantees of Sybil guard are stronger than previous decentralised approaches, they are still rather weak in the absolute sense. Each attack edge allows sybil nodes $O(\sqrt{n} \log n)$ to be accepted. Second, Sybil guard critically relies on the assumption that social networks are fast-mixing, an assumption that had not been validated in the real world”.

The research presents a new protocol that leverages the same insight as Sybil guard, but offers dramatically improved and near-optimal guarantees. The protocol is called Sybil limit because, it limits the number of sybil nodes accepted. It is near-optimal and thus pushes the approach to the limit. Sybil limit achieves these improvements without compromising on other properties as compared to Sybil guard. For example, it provides guarantees on the fraction of honest nodes accepted.

Moreover, their research also considered whether real-world social networks are sufficiently fast-mixing for protocols like sybil guard and sybil limit. Even though some simple synthetic social network models have been shown, to be fast-mixing under specific parameters, whether real-world social networks are indeed fast-mixing is controversial. The authors adopt the premise that social networks are well-known to have groups or communities where intragroup edges are much denser than intergroup edges. Such characteristics, on the surface, could very well prevent fast-mixing. To resolve this question, "experiments with three large-scale (up to nearly a million nodes) real-world social network datasets crawled from www.friendster.com, www.livejournal.com, and dblp.uni-trier.de, found that despite the existence of social communities, even social networks of such large scales tend to mix well within a rather small number of hops (10 to 20 hops)". Sybil limit was quite effective at defending against sybil attacks based on such networks. These results provided the first evidence that real-world social networks are indeed fast-mixing. As such, they validate the fundamental assumption behind the direction of leveraging social networks to limit sybil attacks.

Sybil limit is a near-optimal defense against sybil attacks using social networks. Compared to their previous sybil guard protocol that accepted $O(\sqrt{n} \log n)$ sybil nodes per attack edge, while sybil limit accepts only $O(\log n)$ sybil nodes per attack edge. Furthermore, sybil limit provides this guarantee even when the number of attack edges grows to $O(n / \log n)$. Sybil limit's improvement derives from a combination of multiple novel techniques such as leveraging multiple independent instances of the random route protocol to perform many short random routes, exploiting intersections on edges instead of nodes, using the novel balance condition to deal with escaping tails of the verifier and using the novel benchmarking technique to safely estimate. Finally, their new research results on real-world social networks confirmed their fast-mixing property and, thus, validated the fundamental assumption behind sybil limit's (and sybil guard's) approach.

2.6 Sybil guard

Haifeng Yu et al's previous paper from 2008 presents sybil guard, a novel protocol for limiting the corruptive influences of sybil attacks. It is based on the "social network" among user identities, where an edge between two identities indicates a human-established trust relationship. Malicious users can create many identities but few trust relationships. Thus, there is a disproportionately small 'cut' in the graph between the sybil nodes and the honest nodes. Sybil guard exploits this property to bind the number of identities a malicious user can create. The researchers proved the effectiveness of sybil guard both analytically and experimentally.

Most designs against malicious behaviour rely on the assumption that a certain fraction of the nodes in the system are honest. Sybil guard limits the corruptive influence of sybil attacks, including sybil attacks exploiting IP harvesting and even some sybil attacks launched from botnets outside the system.

Sybil guard is completely decentralised and all functions are with respect to a given node. It guarantees, with high probability, that an honest node accepts at most sybil nodes, whereas the number of attack edges in the system and is the length of the protocol's random walks. Sybil guard achieves this guarantee by bounding the number of sybil groups and bounding the size of each sybil group. Sybil guard bounds the number and the size of sybil groups without necessarily knowing which groups are sybil.

This research provides an overview of a decentralised protocol, with results that show honest node mix up fast among other. Sybil node creates number of false nodes but does not always use them all.

2.7 Decentralised defense, sybil attacks, friends and foes and social ties

In a research paper by Daniele Quercia and Stephen Hailes (2010), the authors proposed a novel decentralised defense for portable devices, called MobID. The aim of their research is to manage two small networks that store information about the devices it meets. Its network of friends contain the honest devices, and its networks of foes contain the suspicious devices. With the help of these two networks, the device is then able to determine whether the an unknown individual is carrying out a sybil attack or not.

The problem considered was, "collaborative applications are easily disrupted by uncooperative and malicious individuals. Those individuals profit from services without providing an adequate return and then make themselves untraceable by creating a very large number of bogus identities."

They found that existing research focussed mostly on peer-to-peer networks. The research in this gap based on two contributed find first, sybil users between friends and foes using MobID guarantees that honest individuals both reject bogus identities and accept honest identities, and they do so with high probability. Second, the robustness of MobID on real mobility and social network data.

"MobID is a protection device mechanism in-range portable devices among sybil attackers with high probability. The research was evaluated using real mobility and social network data and versions of MobID that used k-means clustering were validated as the best performers to protect against attackers who had subverted 20% of real mobile communities without causing any disruption".

In another study B S Jyothi and D Janakiram (2011) proposed SyMon: a practical approach to defending large-structured peer-to-peer (P2P) systems against sybil attacks. SyMon is entrusted with the responsibility of moderating the transactions involving a given peer, and hence makes it almost impossible for sybils to compromise the system. Its effectiveness was proven both analytically and experimentally in large-structured P2P systems.

In another similar research, evolution of person-to-person social relationships, quantify and predict the strength of social ties based on call-detail records in mobile phones. Zhang H. et al (2010), proposed an affinity model for quantifying social-tie strengths in which a reciprocity index is integrated to measure the level of reciprocity between users and their communication partners.

Their research centred on quantifying individual social-ties using a probability model, affinity, in which the reciprocity index is based on mobile phone call records. Affinity is used to measure the similarity between probability distributions, and to quantify the strength of social ties between actors in groups. Taking social relationships into consideration changed the research focus and the strength of social-ties became a function of time. Map call-log data was applied to a time series of the social-ties and the ARIMA model was used to predict social-tie strengths. This study differs from their previous work on measurements, which focused on general structures of social networks where the strength of social-ties in were not considered to be functions of time. However, in the real-world social relationship between two people changes over time.

Observing data in a bi-dimensional matrix, where rows describe data units and columns describe categorical variables. Their calculation for quantifying the strength of social ties is based on the below equation:

$$S_N = \{P = (p_1, p_2 \dots, p_N) | p_1 \geq 0, \sum_{i=1}^{N} p_i = 1\}$$

Further, the research used SARIMA prediction models to identify, estimate and validate predictions in terms of either in-sample or out-of-sample. Their experimental outcomes showed the model to be effective for spam and marketing detection.

In further research, Huiqi Zhang et al (2011), proposed a socioscopic model for social-network and human-behaviour analysis, based on mobile phone call records. Researchers used multiple probability and statistical methods to quantify social groups, relationships, and communication patterns and for detecting human-behaviour changes. The approach included a new index to measure the level of reciprocity between users and their communication partners.

The socioscope comprises several components, including zoom, scale, and analysis tools, which are used for: analysing network structures; discovering social groups and events; and quantifying relationships. Its is extensible, meaning new tools can be added as needed. Zooming-in allows analysts to use multiple scales to investigate social-group member behaviour up to a one-to-one relationship. By zooming-out, general social-network structures and properties can be analysed.

This paper, focused on quantifying individual social groups at the one-to-one relationship level using a probability model, affinity. which It is virtual distance, based on mobile call records and uses a technique as Zhang and their team research above including wavelet denoising, sequential detection, inhomogeneous Poisson and inhomogeneous exponential model, and Bayesian inference for quantifying social groups, relationships, and communication patterns and for detecting human-behaviour changes. The proposed research is based on social network structures and the relationships among the people in the network, along with event detection based on their call records.

Hao Xu et al (2013) state that, “temporary connections cannot be recognised as a real relationship when the history of connections among the nodes are considered. Cumulative stable contacts are proposed to depict the correlation among the nodes. At each timestamp, community cores are detected due to the variation of nodes and links.” The proposed research method effectively detects stable communities in mobile social networks. Introducing the preliminaries used in the paper, their research explores the character of cumulative stable contacts and presents a community detection and tracking algorithm.

They compare their NMI algorithm with COPRA, increasing delta. A few nodes are clustered in community core nodes and many nodes become non-core nodes. The topology of this community is sensitive due to the temporary links caused variations in NMI values. Further, the NMI values of the community detected by CoVa is more stable than COPRA.

In Xiaohui Liang et al’s (2014) research on fully anonymous profile matching in MSN, the authors study user profile matching with privacy-preservation in MSN, and introduced a family of novel profile matching protocols. They first propose an explicit comparison-based profile matching protocol (eCPM) which runs between two parties – an initiator and a responder. An implicit comparison-based matching protocol (iCPM) allows the initiator to directly obtain some messages, instead of the comparison result from the responder. Generalising the iCPM to an implicit predicate-based profile matching protocol (iPPM) allows for complex comparison criteria, spanning multiple attributes. The anonymity analysis shows all these protocols preserve the confidentiality of user profiles.

"A profile matching protocol achieves full anonymity if, after attackers execute multiple runs of the protocol with a user, the probability of correctly guessing the profile of the user is always 1/v", the paper states.

Comparison results reveal the initiator considering k-anonymity as a user requirements which was analysed anonymity risk level in relation to the pseudonym change for consecutive eCPM runs and further introduced an enhanced version of the eCPM, called eCPM+, by exploiting a prediction-based strategy and adopting the pre-adaptive pseudonym change. The defined iCPM and the iPPM both enable users to anonymously request messages and respond to requests according to the profile matching result, without disclosing any profile information.

All the various research above, points to social ties and their connectivity behaviour as the greatest considerations when designing model identify sybil connections using decision trees and a random forest. Their qualifying relationship measure for short and long periods of time contributed in the third and fourth chapter of our research in drawing random charts for sybil detection.

2.8 Random-walks: the evolution of sybil defense protocols

The evolution of sybil defense protocols have leveraged the structural properties of the social graph, with an underlying distributed system, to identify sybil identities. Alvisi L. and his team first clarified the deep connection between sybil defense and the theory of random walks which led to a community detection algorithm that, for the first time, offered provable guarantees in the context of sybil defense. The team advocated a new goal for sybil defense to address the more limited, but practically useful, goal of securely white-listing a local region of the graph (Alvisi L. et. al., 2013).

The first part of their research explains and examines the promise and fundamental limits of a universal sybil defence. At the core of this approach are a set of assumptions about the structure of a social graph under sybil attacks that, essentially, amount to modelling the social graph as two sparsely connected regions -one comprised of sybil nodes and the other of honest nodes consistently connected each other.

Experiments in this research shows, “This model over-simplifies the social structure of the honest region of the graph: rather than homogeneous, this region appears as a collection of tightly-knit local communities relatively loosely coupled with one another.”

The second part of their research advocates a realignment of the focus of sybil defense to effectively leverage the strength of communities against sybil infiltration. Research problem about casting sybil defence and checking that done off the shelf community detection really possible. Contributing two part of for this research. This approach requires extreme caution because the choice of a community detection protocol can dramatically affect whether sybil nodes are accepted as honest. The study goes on to outline its mathematical foundation for sybil detection.

The research serves as a warning against the danger of falling into a Maginot syndrome: the building an ever more sophisticated line of defense against attacks that the enemy can easily circumvent. The research arguments shows that, the key addresses these challenges was defined in depth and their past research defence layers and ill-quipped to detect (Alvisi L. et. al., 2013).

The research goes on to examine which of four fundamental structural properties of social graphs would best serve this aim: popularity; small world properties; clustering coefficients; or conductance questioning. The answer they found was conductance, a property intimately related to the concept of mixing the time of a random walk. The next section proposed and discussed protocols to exploit variations in conductance as a basis for a decentralised universal sybil defense. In bad cases protocols become vulnerable to sybil attacks, which further lead to observations of social graphs that have an internal structure organised around tightly knit communities and the graph properties crucial for sybil defense are significantly more likely to hold within a community rather than in the entire social graph (Alvisi L. et. al., 2013). Reviewed provides a solid theoretical foundation to sybil defense based on community detection brought and showed how the well-known concept of Personalised PageRank offers honest nodes a path towards a realistic target for sybil defense that is more limited than universal coverage.

The research concludes with outcomes and shows that sybil defense protocols based on random walks continue to be effective when used in combination with very simple checks that leverage the structural properties of the social graph other than conductance (Alvisi L. et. al., 2013).

This research brought the viewpoint of random walk and structural properties of social graphs which have need considered in our third chapter of research. Furthermore, as explained rather than attempting a universal sybil defence, a piecemeal approach we should do it in pieces. So our research follows the classification and regression analysis in two parts and continuing to future work with new analysis procedure of cascading.

2.9 Decision-making processes

Cortimiglia, M. N. et al (2011), presented a case study on an Australian mobile network operator (MNO) to highlight the decision-making process for launching a mobile networking service. They demonstrated that through critical analysis of competitors, customers, and value delivery, it was possible to derive a three-step strategy MNOs could adopt when offering mobile social networking services. Integratation of mobile services into social networks was a key part of the strategy. It would provide enriched and differentiated experiences that would, in turn, drive usage and acquisition of MNO services.

Their research compared a number of service providers and their customers in the Australian mobile market. Competitors, customers and value delivery were jointly analysed to identify the overall strategic approach to the market each company took. Logs kept on each customer for period of observation shed light on how MNOs are used. Cortimiglia, M. N. et al's research explains "social network access to mobile is offered by MON nowadays providing access to sites, increase the traffic increase mobile usage. Another possibility to extend SN access to mobile involve technical solutions for single sign in procedures and seamless usage, thanks to the storing of user login information in MNO's databases or the identification of user SIM cards. Finally, MNOs can provide applications that facilitate posting content created with the mobile devices such as Instagram in SNS. Furthermore, updates and versions of application, MNOs develop and offer their own balancing services and applications that improve users online SNS usage. Lastly, drive usage and acquisition of MNO services with SN integration was the last step in the proposed MNO strategy to profit from mobile social networking, which mainly entails linking users' activities in mobile services with the online SNS world".

Integration between MNOs and SNS providers is bidirectional, which means that the interests and potential benefits accrued by SNS provider are just as important to the MNO's.

Considering points of this research study in our thesis, this research showed behaviour of the service provider, interrelated connection of users and activities which helped us to think about links between users in our research while classifying the sybil in our dataset.

2.10 Social graph mining

Mining (Social) Network Graphs to Detect Random Link Attacks research by Nisheeth Shrivastava and team provided a generic abstraction of spam emails, annoying tele-marketing phone calls, viral marketing in social networks attacks, called the Random Link Attack (RLA) that can be used to describe a large class of attacks in communication networks. The research mine the social networking graph extracted from user interactions in the communication network to find RLAs and formally define RLA and show that the problem of finding an RLA is (theory) NP-complete (Shrivastava N. et. al., 2008).

The first contribution of research was, formalise the notion of the Random Link Attack (RLA) based on simple interconnection properties of individuals in any communication network proving that the problem of finding an RLA was NP complete. Further, presented two simple tests, called the clustering test and the neighbourhood independence test, that was used to quickly mark a small subset of nodes as suspects and prune away the rest of the social networking graph including empirically evaluated the effectiveness of these tests in identifying suspects. Then presented two heuristics, GREEDY and TRWALK to mine subgraphs satisfying the RLA property, starting from the suspect nodes. The techniques performed well in practice for detection (Shrivastava N. et. al., 2008).

Even though, their research concentrated on emails spam and telemarketing call, we conclude that, the method of tracking attacker using RLA made a significant contribution in our research for addressing issues in third chapter. The idea of structuring to identify the attack on social network through graph mining contributed in our research to identify sybil users and their behaviour of connections based on connectivity pattern helps to trace the sybil links in our classification model for sybil detection.

In Another research of mobile social network, Yuhang Zhao and team proposed, the mobile spam is getting increasingly serious, troubling users daily life and ruining the service quality. The researcher Yuhang Zhao and team proposed a novel approach for spam message detection based on mining the underlying social network of SMS activities. Experimental results on real dataset illustrate effectiveness of various features, showed desired results. With the help of the robust structure of social network constructed from SMS, team able to overcome the frailties of traditional methods such as keywords and flow rate filtering (Zhao Y. et. al., 2012).

The approach have proposed and evaluated more graph features, such as maximum clique number and chromatic number, and where it can obtain more details about the social network extracted from SMS behaviour. With the help of ensemble learning, produced classifier for spam detection than the one with only simple features, such as degree features or communication coefficient. Research observed that, even on an incomplete dataset of only 3-day data, with sparse links between nodes, those features does still work well, achieving a precision of 99.9% when the recall value is above 95% (Zhao Y. et. al., 2012).

In our conclusion, research presented using complex graph structure and method filtering data describing network flow and showed effectiveness of those features with several off-the shelf classifiers. This help us in our chapter 3 and even in chapter 4 for graph based and classification technique in for predicting and comparing sybil nodes.

2.11 Security Measures

An effective network security strategy requires identifying threats and then choosing the most effective set of tools to fight them. There are three ways to deal with attacks in network that are Detection, Prevention and Counter Measures which are studied by Ankita S. Koleshwar and research team in the article published by international journal of ARCS. The RSS based detection mechanism was presented for the sybil attack problem in the network. RSS based detection mechanism gives the better result for protecting the network from sybil attack. The research study provides the overview of security measures of different types of attacks on Mobile and cloud computing (Koleshwar A. S. et. al. 2014).

In our conclusion, this past of research provided an overview of security measure for us while detecting sybil on social network and mobile technology which help to identify the pattern and type of attack.

2.12 Sybil attacks and defenses in the Internet of Things (IoT)

Kuan Zhang, Xiaohui Liang, Rongxing Lu and Xuemin (Sherman) Shen, researched on sybil attacks and their defense in the IoT proposed survey sybil attack and defense system in IoT. Their research explained about the types of sybil attacks considering sybil attacker's capabilities. Also, the research presented some sybil defense schemes, with a social graph based sybil detection, behaviour classification based sybil detection and mobile sybil detection with the comprehensive comparisons (Zhang K. et. al., 2013).

The research paper surveyed the sybil attack and the corresponding defense schemes in IoT evaluating three types of sybil attacks SA-1, SA-2 and SA-3 to cover a broad range of sybil attack. SA-1 considering a limited number of connections with ordinary user, SA-2 considered building many such social relationships, and SA-3 were considered in mobiles networks where social graph information is not available and cannot be easily detected. Furthermore, their research presents three types of sybil defense schemes. First social graph based sybil detection (SGSD), second, behaviour classification based on sybil defense (BCSD), and third, mobile sybil defense (MSD) with the comprehensive comparisons including discussion of research issues (Zhang K. et. al., 2013).

Furthermore, in the section of research Social Graph sybil detection expressed the finding based on two assumptions. One, social network base sybil defense (SNSD) is a kind of sybil defense based on social network, which is a social structure linking social relationships among nodes. The research team thinks that sociology theory was a useful tool to investigate the relationship among the users. In this section, the term social network indicates the user's social graph and structure, which can reflect user's social relationships and the social trustworthiness among users. Leveraging the social network structure propose a famous SNSD scheme, Sybil guard, based on a random walk, research gave an assumption as follows:

1: Although the sybil nodes can tightly connect with other sybil ones, the number of social connections among sybil nodes and honest ones is limited. In the conclusion of this assumption, research team explains the trustworthiness or credit can enhance the establishment of the social graph and restrict sybil attackers to build connections with normal users so that the detection accuracy can be improved. Second, Social Community-based sybil Detection (SCSD) explores social community detection to facilitate sybil detection. The possibility of using social community detection algorithms to detect SA-1 was validated with the help of past researchers.

2: Sybil nodes cannot tightly connect with honest nodes in the multiple honest communities since honest nodes would not trust sybil ones. Honest nodes can closely inter-connect with others in the honest community. The researcher compares the SGSD schemes with respect to preliminary techniques, assumption, decentralised properties, etc. A tendency is to explore trustworthiness to facilitate the sybil detection to SA-1.

Moreover, in their section of research on behaviour classification-based sybil defense from a recent study by (Jiang J. and team (2012)), sybil users in RenRen a popular OSN in China can generate an exponential number of social connections with the normal (or honest) users. In Conclusion of this part they found, strong sybil attackers would penetrate into the social graph and generate many social connections with the normal users, which opposes the assumption of the SGSD. If sybil attackers are familiar with the normal users click patterns or habits, sybil attackers could truly mimic the normal users, the BCSD cannot efficiently detect them as well. However, it is obvious that sybil attackers have to consume a large portion of time to mimic the normal users so that the attack behaviours are partially limited.

Besides, research team present the mobile defense schemes that explore, without the global social graph for sybil detection, MSD aims to either detect SA-3 or restrict sybil attacker's behaviours. To continuing this part of research, team surveyed and presented work about the friend relationship based sybil detection, cryptography based mobile sybil detection and Feature-based mobile sybil detection which again considers SA-1, SA-2, and SA-3. The further part of Research Challenges shows the future aim of research in sybil Defense in Mobile social network, Privacy and sybil defense and Cooperative sybil defense.

In my conclusion, this paper researcher has provided a survey of sybil attacks and their defense schemes in IoT. Specifically, the team had defined three types of sybil attacks in the distributed IoT and presented some sybil defense schemes with the comparison. The differential characteristics, including social structures and behaviours between sybil attackers and normal users, could facilitate the sybil defense. Furthermore, the research team found that MSD can leverage mobile network features, wireless channel characteristics, and cryptography to resist sybil attackers. They have also suggested some open research issues such as sybil defense in MSNs, the trade-off between privacy and learning in sybil defense, and cooperative sybil defense (Zhang K. et. al., 2013).

2.13 Exploiting Mobile Social Behaviours for sybil Detection

In another research proposed by Kuan Zhang, Xiaohui Liang, Rongxing Lu, Kan Yang, and Xuemin (Sherman) Shen, in exploiting mobile social behaviours of social detection, explained, Social-based Mobile sybil Detection (SMSD) scheme to detect sybil attackers from their abnormal contacts and pseudonym changing behaviours. In research, the process flow as: firstly, "by defining four levels of sybil attackers in mobile environments according to their attacking capabilities. Secondly, exploit mobile user's contacts and their pseudonym changing behaviours to distinguish sybil attackers from normal users. Utilising a ring structure associated with mobile user's contact signatures to resist the contact forgery by mobile users and cloud servers. In addition, investigating mobile user's contact distribution and social proximity, proposed a semi-supervised learning with Hidden Markov Model to detect the colluded mobile users. Security analysis demonstrates that the SMSD can resist the sybil attackers from the defined four levels, and the extensive trace-driven simulation shows that the SMSD can detect these sybil attackers with high accuracy." (Zhang K. et. al., 2015).

Sybil attackers frequently change their pseudonyms to cheat other users. Researcher investigated the contact statistics of the used pseudonyms and detected sybil attackers by comparing the contact statistics of pseudonyms from normal users and that from sybil attackers.

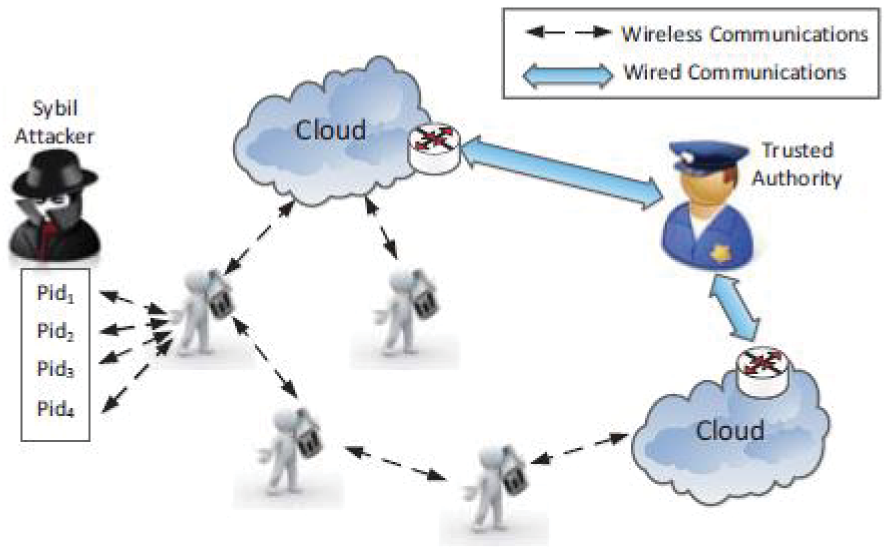


Fig. 2.2 Research System Model (Zhang K. et. al., 2015)

The System model and attacker model contains following:

The system model contains 3 entities. One trusted authority (TA) which bootstraps the whole system and generate secret key to mobile users. Two, Mobile Users have bi-directional communication with each other. User should be register to TA. So the user adopts pseudonyms and prevents its real identity. Third, Cloud Server (CS) is semi-trusted entity in system which has powerful storage and computing capability which can directly communicate to mobile users and collect their data.

The Attacker Model defined in 4 level. First, General sybil Attackers (Level-1): sybil attackers, denoted as *As*, exist in mobile environments to compromise the normal user and adopt the pseudonyms to hide identity and repeatedly send similar information to users. Second, sybil Attackers with Forged Contact (Level-2), A sybil attacker *As* would forge his contact information to benefit *As* during sybil detection. Many contact can provide record against *As* evidence of changing pseudonyms frequently. Third, sybil Attackers with Mobile User's Collusion (Level-3). In which, mobile users may collude with sybil attackers to illegally provide fake contact information and disrupt sybil detection. It can generate valid signature on inexistent contact with sybil attacker even unknown for each other. Four, sybil Attackers with Collusion of Cloud Servers (Level- 4). Even though the CS can honestly follow the protocol, but it is a semi-trusted entity. If a CS is compromised or colludes with the sybil attacker *As*, the CS could either add some fake contact information for *A*s, or modify and delete the normal user's contact information to increase the false detection rate (Zhang K. et. al., 2015).

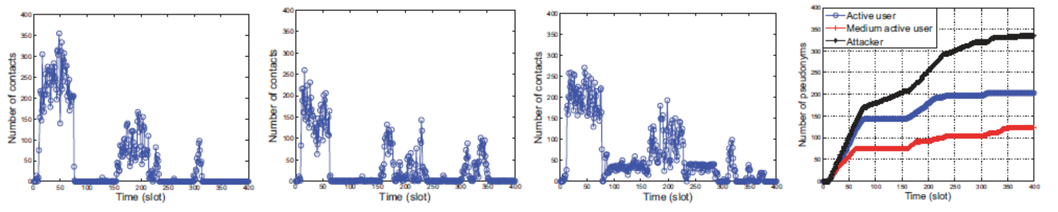
Further, research defines the design goals toward achievements.

1) General Mobile sybil Detection specifies, when a Level-1 sybil attack maliciously changes his pseudonyms to launch attacks, the scheme should detect this from his mobile social behaviours.

2) Unforgeability: The proposed scheme should be able to prevent attackers from forging the contact information, since the forged contact would disrupt the sybil detection. The encountered users should exchange unforgeable information (e.g., signatures of the contact) to the other user, and keep the integrity of contact information.

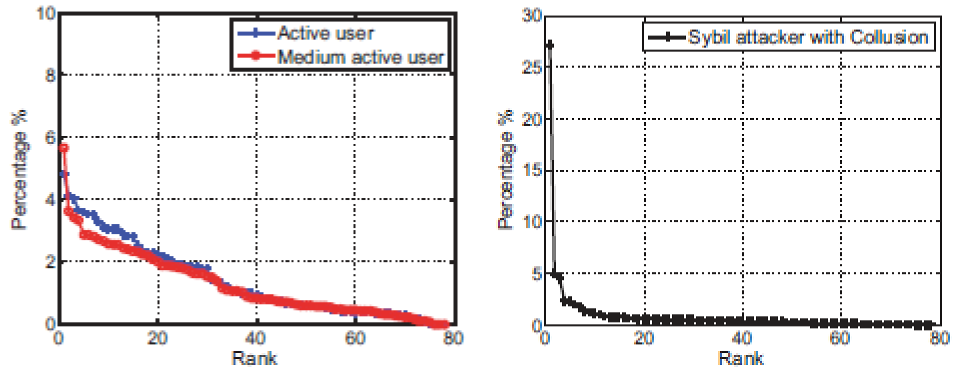
3) Resistance to Collusion of Mobile Users: The proposed scheme should be able to resist the collusion of mobile users. It is critical to find out the forged inexistent contacts when mobile users collude.

4) Resistance to Collusion of Cloud Servers: The data stored in the cloud server should not be added, modified or deleted by CSs (i.e., Level-4 attackers). The modified data should be detected by other entities, e.g., the TA and mobile users.



(a) Contact of an active user (b) Contact of a medium active user (c) Contact of an attacker (d) Pseudonym changing among normal users and Sybil attackers

Fig. 2.4: Observation on contact information and pseudonyms changing between normal users and sybil attacker (Zhang K. et. al., 2015).



(a) Contact rate distribution of normal users (b) Contact rate distribution of the Sybil attacker

Fig. 2.5: Contact rate distribution charts (Zhang K. et. al., 2015)

Finally, the research performance analysis shows that Infocom06 trace with 78 mobile users during for day conference assigning users with social communities according to the sociology theory. A complete graph G was built up, where each edge weighted by the total number of contacts between two vertex. We then refine the graph with 78 vertices and 2, 863 edges by removing the edges weighted less than 100. Based on Bron-Kerbosch algorithm, they extract maximal cliques in G selecting 100 social communities (i.e., cliques). The selected communities are used for simulation comparison on social connections (Zhang K. et. al., 2015).

At end research compared the SMSD and LSMSD schemes with the sybil detection scheme which is in D. Quercia and S. Hailes research, named FFL (Friend and Foe list). In FFL, mobile users detect attackers by comparing their social friend list and determining the sybil attackers similarly, and selected 1, 4, 8, 12, and 16 users as the attackers in our simulation. Further utilise false positive rate (FPR) and false negative rate (FNR) as the metrics to evaluate the sybil detection accuracy. A false positive detection results in a normal user being detected as an attacker, while a false negative indicates that a sybil attacker being regarded as a normal user (Zhang K. et. al., 2015).

In our conclusion, research presented, experiments and evaluated their specified goals for detection of sybil users. This research specifically taken in to consideration as we are using same dataset with updated version of 98 users. Their results compared at the end of chapter 4 and drown extra sybil user from our newly designed analytical and modelling techniques.

2.14 Other researches view points

Research present two forwarding protocols for mobile wireless networks of selfish individuals. Extensive simulations with real traces show that protocols introduce an extremely small overhead in terms of delay, while the techniques introduce in research help to force faithful behaviour have the positive and quite surprising side effect to improve performance by reducing the number of replicas and the storage requirements. The protocols also tested in the presence of a natural variation of the notion of selfishness - nodes that are selfish with outsiders and faithful with people from the same community. Even in this case, protocols are shown to be very efficient in detecting possible misbehaviour (Mei A. et. al., 2013).

Encounter-based social networks and encounter-based systems link users who share a location at the same time, as opposed to the traditional social network paradigm of linking users who have an offline friendship. This new approach presents challenges that are fundamentally different from those tackled by previous social network designs (Mohaien A. et. al., 2013)

Web transaction data usually convey user task-oriented behaviour pattern. Web usage mining technique is able to capture such informative knowledge about user task pattern from usage data. With the discovered usage pattern information, it is possible to recommend Web user more preferred content or customised presentation according to the derived task preference. The proposed research on a Web recommendation framework is based on discovering task-oriented usage pattern with Probabilistic Latent Semantic Analysis (PLSA) model. Combining the identified task preference of current user with the discovered usage-based Web page categories, research present user more potentially interested or preferred Web content (Xu. G. et. al., 2006).

Research paper Enabling Trustworthy Service Evaluation in Service-Oriented Mobile Social Networks published by Xiaohui Liang and team identify three unique service review attacks in propose a Trustworthy Service Evaluation (TSE) system to enable users to share service reviews inservice-oriented mobile social networks (S-MSNs), i.e., linkability, rejection, and modification attacks, and develop sophisticated security mechanisms for the TSE to deal with these attacks with extension of the bTSE to a sybil-resisted TSE (SrTSE) to enable the detection of two typical sybil attacks (Liang X. et. al., 2014).

2.15 Summary

The chapter introduce the sybil and related work done so far in analytics, mobile social network, and crime and investigation area. Section 1 provided

Defense against sybil Attack in Directed Social Network in which research presented algorithm based on the set of measures and iterative optimisation to detect the sybil region. The research explained about social ties and Social relations in microblogging services are very different from that in undirected social networks. Comparison also provided with sybil defender shows that their proposed algorithm is superior for the sybil region detection problem in directed social networks. Section 2 introduce the about defending sybil attacks in MSN Yan Sun and team proposed a security mechanism to detect sybil nodes. Section 3 provide same topics of research in which many researches are contributed in area looking after different finding and explaining different modelling techniques. This research proposed by Chang W. and research team explained the local ranking system for estimating trust level between users. Their designed system uses a distributed sybil defense algorithm based on signed networks. Further in same section, Jiang J. and team produced a research on detecting and validated the sybil groups in online social network in RenRen. Section 4, 5 and 6 explain the mechanisms of sybil defender, sybil limit and sybil guard explain different methods and protocols and contributing area of sybil detection. Section 7, whereas combined and explain many researcher work done based on decentralised defense, sybil Attacks, Friends and Foes and Social Ties on mobile social network. Section 8, based on random-walk evolution of sybil defense protocols explained by Alvisi L. and researcher team. The research flows examining four fundamental structural properties of social graphs based on popularity, small world property, clustering coefficient, and conductance questioning which can serve better. Section 9, explain the decision making process in a case study in an Australian mobile network operator (MON). By describing competitive analysis, customer analysis and value deliver analysis shredded the light on usages of MNOs and SNS. Section 10 brought the importance of social graph mining which contributed in our research chapter 3. Section 11, explain about the security measures of identifying threats and then choosing the most effective set of tools to fight. Section 12 and 13 provided the knowledge about the sybil attack and their defences and sybil behaviour in MSN. These researches played important role and contributed to our research to identify sybil users. Using classification technique we compare their results with us in over the chapter 4. Section 14 provided the other views from different research areas to our research, in identification and behaviour of sybil user at potential level. Some other and more researches have been studied and acknowledged in bibliography at the end.

Chapter 3
Graph-based Behaviour Analysis using Connectivity between Nodes

3.1 Background

Users of mobile technology and apps are quickly occupying the personalised services market. Uploading data, use of social networking, banking systems, storage and production of text data, images, animations and storage of confidential information on mobile has become much easier than it was on computers and laptops. Spreading and transferring such data through mobiles and social networks is also easier than via email. Mobile technology's ability to save time and its high appeal to human psychology and behaviour have been recognised (Cortimiglia, M. N. et al's, 2011).

Additionally, data compression has become easier due to smart phone technology, as a free and quick alternative to computer software. Free apps and inbuilt mobile technology have made our social, financial, business and cultural lives easier than before.

However, this technology has also simplified the work of hackers and attackers, helping them to steal personal information in a fraction of a second, and without the need for any special software. Such attackers are called a sybil attackers on mobile social networks. Sybils are users who pretend to be an honest entity to collect information with intention of blackmail, money laundering, hacking, or to perform other criminal and offensive activities. The data may be taken for their own use or to distribute amongst other sybil attackers.

Honest nodes, or users, are authentic nodes connected to other honest nodes. Sybil nodes, or users, are those using a false identity designed to steal information. False nodes can also be connected to other false nodes for the purposes of exchanging data collected between sybil users, and this behaviour can be tracked.

Sybil users are those who create a false identity on mobile social network for the intention of data distribution. A single user or a group of sybils may intend to harm the social media platform, a particular organisation, or an individual. The intention behind their activity can range from: reducing the revenue of an organisation; stealing customer data to sell to a competitor; blocking or hampering the social media provider and their services to decrease user involvement or new user registrations; or to misuse an individual's data to a achieve personal objective; along with many others.

Honest users access their identity on mobile social networks for the purposes of personal communication. An honest entity may connect one or more trustworthy entities including few, as yet unknown, sybil identities. Conversely, a sybil user may have more than one identity on mobile social network. These identities might be accessed from one mobile device or multiple devices. The purpose of this is, again, to interfere with honest entities and their information.

Two or more sybil users may collude to form a group or community of sybils, whose purpose is to generate high levels of destruction or harm to an individual or organisation. These attacks are executed by sybil users pretending to be honest users within an honest community. One or more sybil users may involve themselves in an honest community, by pretending to be an honest user and then passing the information to the other sybil users, or the sybil community or to the honest community. The variety of profiles this creates can make it hard to distinguish the sybil users from the honest ones. Experiments based on connection analysis and content will evaluate this research outcome, using more assumptions, classification and regression techniques, and decision trees for substantial proof.

Little research has been published on identifying the abnormal behaviour of sybil users, yet the intention to cause harm is common in human psychology. This kind of behaviour may result from a mind-disturbing event in one's past, as in the case of the real life Sybil, or may be just for fun. In real life, this type of abnormal behaviour can be discerned. For example, when a couple breaks up or when a person launders money. Sybil users perform similar actions in the virtual world.

3.1.1 Scenarios

Abnormal behaviour can be observed in the following scenarios: the break-up of a couple, resisting attack, money laundering, stealing individual information for personal use, performing a terrorist attack, collecting data from multiple users to sell to other organisations to generate revenue, reducing or descripting revenue to an organisation, harm to an individual's life or organisation for personal reasons, based on an prior incident, or it could result from many human psychological barriers. Such behaviour ranges from low intensity, such as stealing data from an individual, to very high intensity, such as a terrorist attack in the real world using social media as its communication channel. Similarly, some behaviour can be seen as childish teasing, or born of ignorance or a poor education.

This research aims to identify sybil users or groups of sybil users in a complicated environment. The study evaluates a data set to find connections among its users that distinguish suspicious activity performed by false identities to improve social media, and reduce its threats to individuals and organisations.

Many information sources within the domain were reviewed to enrich the quality of this research to ensure it plays a significant role in social media analytics. The research proposed in this paper resolves the following issues:

1) Does abnormal behaviour between nodes on mobile social networks distinguish connections between honest and sybil nodes?
2) Do irregular times of connection between nodes play a vital role in identifying this abnormal behaviour?
3) Does the connectivity intensity or frequency of connections help to identify sybil users?

Taking dataset availability and the records within it into consideration, the above research problems can be evaluated using the Infocom06 dataset. This dataset contains User1 connection (total 98 users) and connection to User2 (over 4000 users) connected multiple times to each other. Each connection records a start time, an end time, its index value and connection id.

Final outcomes are explained in the form of algorithm, graphs, charts and a table that displays information about each node including assumptions, its sybil category, the reasons for its classification as sybil, and each node connection to other nodes.

3.1.2 Research process for identification

This chapter aims to use the proposed graph-based system to determine the connectivity intensity, or the number of times an honest node connects to a sybil node, and will demonstrate that sybil attacks can be traced with the help of graphs and a behaviour analytics system.

To identify sybil users, amongst honest users, a cross-checking method examines the number of times a connection occurs with a single user or multiple users. By way of practical explanation, first, suspicious activity by false identities was determined by at least one, or a regular, or a continuous connection with single or multiple users for a longer than usual time. This helps to identify sybil users. Second, an irregular time of connection between two nodes creates suspicious activity between nodes. For programming this algorithm, MATLAB and SPSS were used to develop the graph-based system that helps to identify the connectivity intensity between: honest nodes and sybil nodes; and between sybil nodes and other sybil nodes.

Significance 1: Identification of false identities using a graph-based technique. Using graph-based techniques in this research the User 1, User 2, starting time and ending time of the dataset infocom06 will produce the different graphs and charts to identify sybil users. It also evaluates the relationship between nodes based on the number of connections between each user. Further, this graphs help to identify sybil attackers using the behaviour of the node.

Significance 2: Ability to track the behaviour of sybil nodes

Using behavioural techniques, the research outcomes demonstrate that sybil users can be identified based on the duration of connectivity between each user – from the start to end time of each connection, and the number of connections with a single user. These results complement Significance 1 and help achieve better success in tracing sybil users in the dataset.

3.1.3 Graph theory

Graph theory and recently developed graph systems have been widely adopted in network analysis, and are partly responsible for the outcomes of this research. However, many other disciplines must be integrated to achieve the goal of finding sybil users in honest communities. Behaviour analytics helps trace the behavior of sybil users in both honest and sybil regions. Social network-based sybil defense systems contribute to identifying sybil defenses that suffer high false positive rates. Neighbour monitoring helps identify authentic and false identities in local stored regions. Signed social network systems help determine trustful relationships between users, but more importantly the distrustful ones. Performance analysis and evaluation systems help measure impacts on honest communities, again with the assistance of graphs. And the list goes on.

3.2 Research methodology

3.2.1 Objectives

Objective 1: Identify the behaviour between nodes on mobile social networks using their connectivity.

A sybil entity can be recognised by the connectivity between a sybil user and other honest users using a cross-checking method. This could include the number of connections made by a single user or multiple users. Suspicious identities are characterised by a higher number of connections between individual entities. On the other hand, most sybil attackers behave in a similar way to normal users which sometime hard to identify. They share information with other sybils a number of times. Multiple connections with a single user could help distinguish a sybil user.

Objective 2: Identify a specific link between an honest node and a sybil node using the duration of connection as an attribute.

Sybil entities can be traced in two ways. First, authentic relationships with users last for a longer period of time, which helps to identify sybil users. Identifying the connection time between two nodes also helps to separate sybil users from honest ones. Second, irregular connection times between two nodes indicates suspicious activity. These intermittent connections could be due to the time period, abnormal time of connection, or using a different location and IP address each time.

Objective 3: Development of an algorithm supported by a graph-based system to identify the connectivity intensity between nodes.
When the connection changes, the frequency of the connection also changes including node dependencies. Each node has a different intensity, or frequency of connection to other nodes. Similarities between connections and their duration defines the intensity level of each node, which helps to categorise nodes, predict their behaviour and draw conclusions between honest and suspicious activity.

3.2.2 Processing method

The graph-based system and behaviour analytics results were produced using MATLAB and SPSS.

Step 1: Generate a graph using MATLAB and SPSS, and build a relationship between User 1 and User 2 in the Infocom06 dataset to achieve Objective 1.

Many relationships among users have explicit linkages. The most expedient method of beginning the research was to evaluate the dataset by plotting some random graphs that consider the relationships amongst users.

The bar chart in Figure 3.1, shows the maximum number of connections User 1 made to each user along the x-axis (User1 in the dataset), and number of those connections along the y-axis (User2 in dataset). Given the chart demonstrates both the highest and the lowest connectivity between one user and all the others; it helps identify sybil nodes because they typically show higher connectivity.

The graph also expresses the user's average number of connections to the others, and is considered to be an honest node, at this stage. Most honest nodes have the same average number of connections to other nodes.

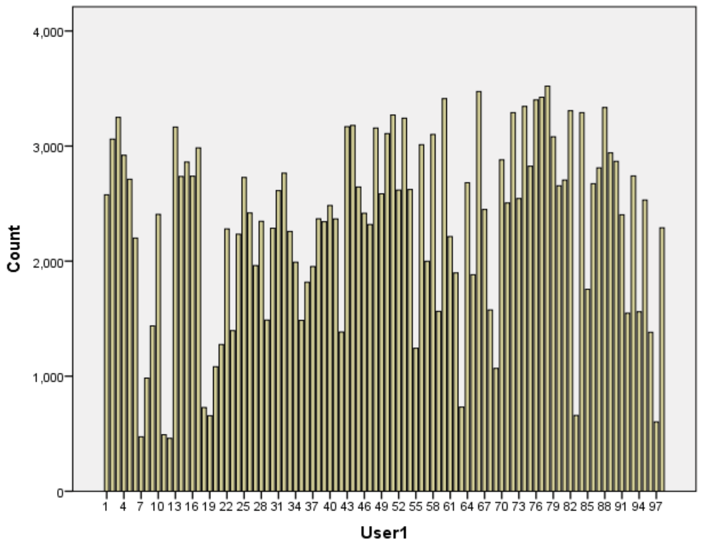


Figure 3.1: Count of connections between User 1 and all other users in the Infocom06 dataset

Step 2: Generate a graph using SPSS and produce a relationship between User 2 and User 1 of the Infocom06 dataset to achieve Objective 1.

Step 1 is now flipped and the opposite result is produced to cross-check user connectivity, and the connectivity of each user to User 1 was graphed. Figure 3.2provided a crucial outcome. Connectivity between users is very high, and gradually reduces until the last user of User2. This graph contributes to the intensity of the user connections.

Algorithm 1 produced both (Figure 3.1 and 3.2) graphs as follows: (Note: this algorithm may change with further research progress.)

```
Algorithm 1(as Depicted in Figure 3.1)
Algorithm to plot the Attribute ID vs Quantity (highest connection) of users connected to the User ID Graph.
```

```
Procedure: Attribute ID vs Quantity of user
X = Dataset Column 1 // Attribute ID
Y = Dataset Column 2 // Max number of users connected to Attribute
ID
Assign a, b, c as integer
Variable c set to 1
While X <= Xmax do
    Set c to 0
While Y <= Ymax do
Assign X to a
      Assign Y to b
       Increment X by one// (row =row +1)
      Check condition if (X == a) AND (Y == b) THEN
```

```
        Increment X by one
 Else
    Increment c by one // User Counter
      End if
     End while
PLOT (Dataset, X, c)   // plot single bar on graph
Increment X by one
End While
End Procedure
```

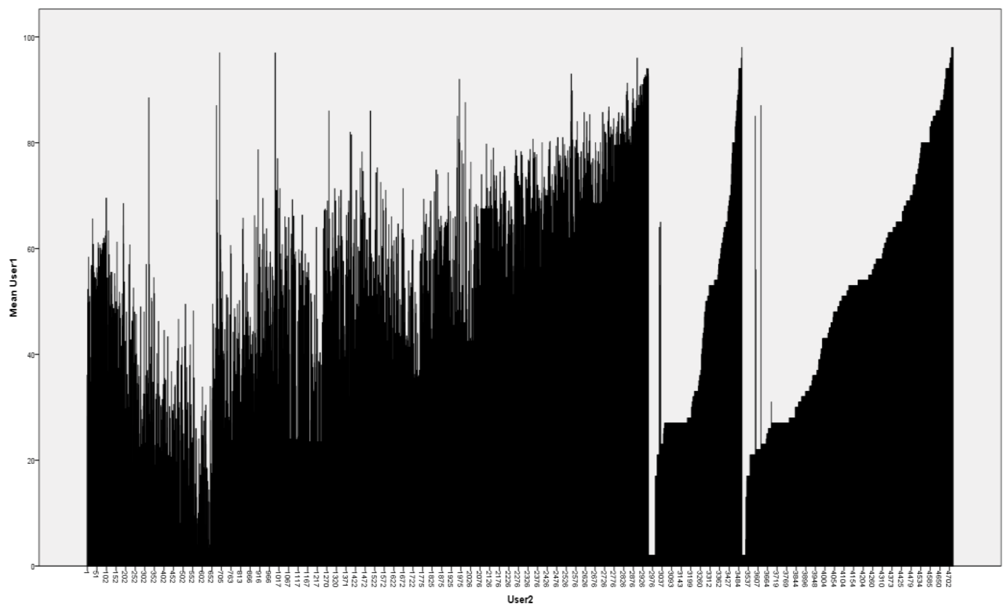


Figure 3.2: User 2 vs User 1 suspicious identity with high and low connections

Definition: Maximum number of connections

$$\mathrm{Hn} = \text{Honest entity}$$

$$\mathrm{Sn} = \text{Sybil entity}$$

$$\lim \mathrm{Hn} \rightarrow \partial$$

$$0 < \partial < 300$$

$$\lim \mathrm{Sn} \rightarrow \partial > 300$$

```
Algorithm 2 (as depicted in Figure 3.2)
Algorithm to plot the User ID vs the Quantity of Attribute ID
graph.
```

```
Procedure: User ID vs Quantity of Attribute ID
X = Dataset Column 1
Y = Dataset Column 2
Assign a, b, c as integer
```

```
Variable c set to 1
While X <= Xmax do
    Set c to 0
While Y <= Ymax do
      Check condition if (X == a) AND (Y == b) then
        Increment X  by one
        Increment c by one // User Counter
End if
If X is not equal to a then
    PLOT (Dataset, Y-axis, c) // plot single bar on graph
    set c to 0
End if
    End while
Increment X by one // X-axis row
End while
End procedure
```

```
Algorithm 3 (as depicted in Figure 3.3)
Algorithm to plot User ID vs High short-time connectivity with the
User ID Graph.
```

```
Procedure: User ID vs High short-time connectivity
X = Dataset Row 1, Column 1
Y = Dataset Row1, Column 2
Assign a, b, c as integer
Variable T1 = Dataset Row1, Column3
Variable T2= Dataset Row2, Column 4
While X <= Xmax do    //Outer loop
    Assign X to a
      Assign Y to b
    While Y <= Ymax do // Inner loop
      Check condition If T1 Same as T2 then
        Increment c by one // User Counter
End if
Increment X  by one
Variable d set to Y
If b is not equal to d then
PLOT (dataset X, c) // plot single bar on graph
    Set c to 0
End if
    End while
End while
End procedure
```

Definition: Short time connectivity

$$\lim \mathrm{Hn} \rightarrow \partial$$

$$0 < \partial < 1\ sec$$

$$\lim \mathrm{Sn} \rightarrow \partial$$

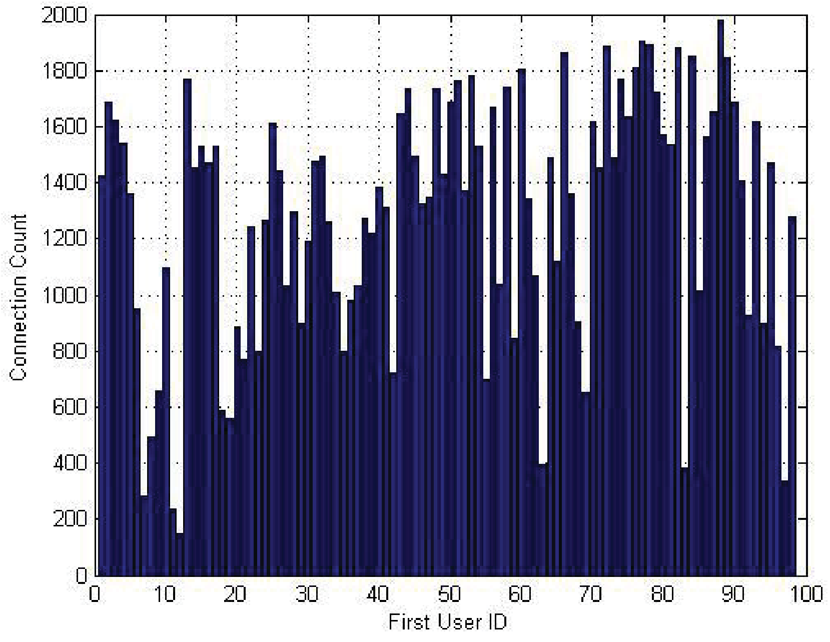


Figure 3.3 Bar graph: User 1 vs high short-time connectivity between various connections

Other results produced using MATLAB and SPSS are as follows:

Nodes ID	Number of Nodes
88	1978
77	1901
78	1889
72	1884
82	1883
66	1862
84	1848
89	1846
76	1809
60	1804

Table 3.1: User 1 vs short-time Count

Nodes ID	Connection count
27	~800
69	~400
2	~350

Table 3.2: Attribute ID vs Quantity (highest connection)

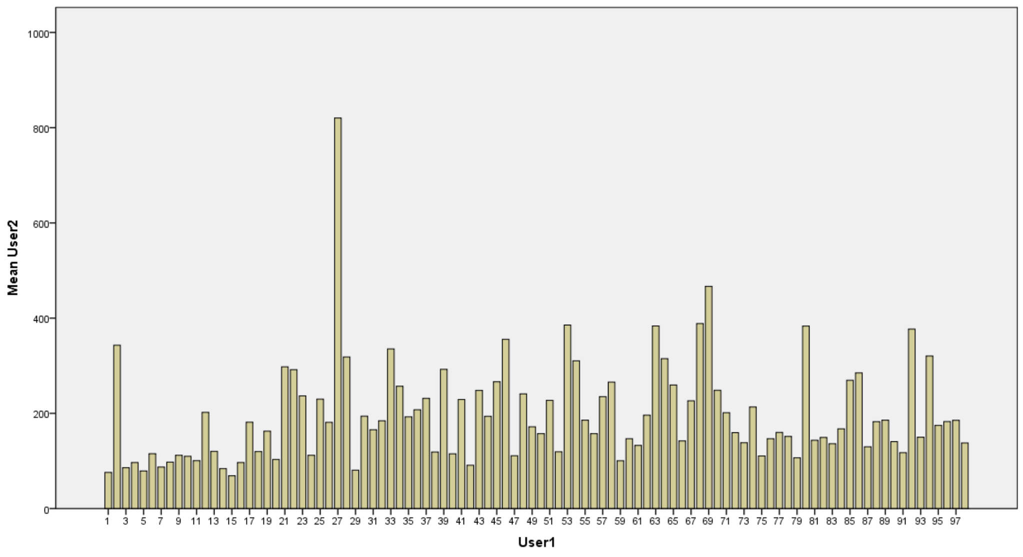


Figure 3.4: Highest connecting User 1 vs User 2

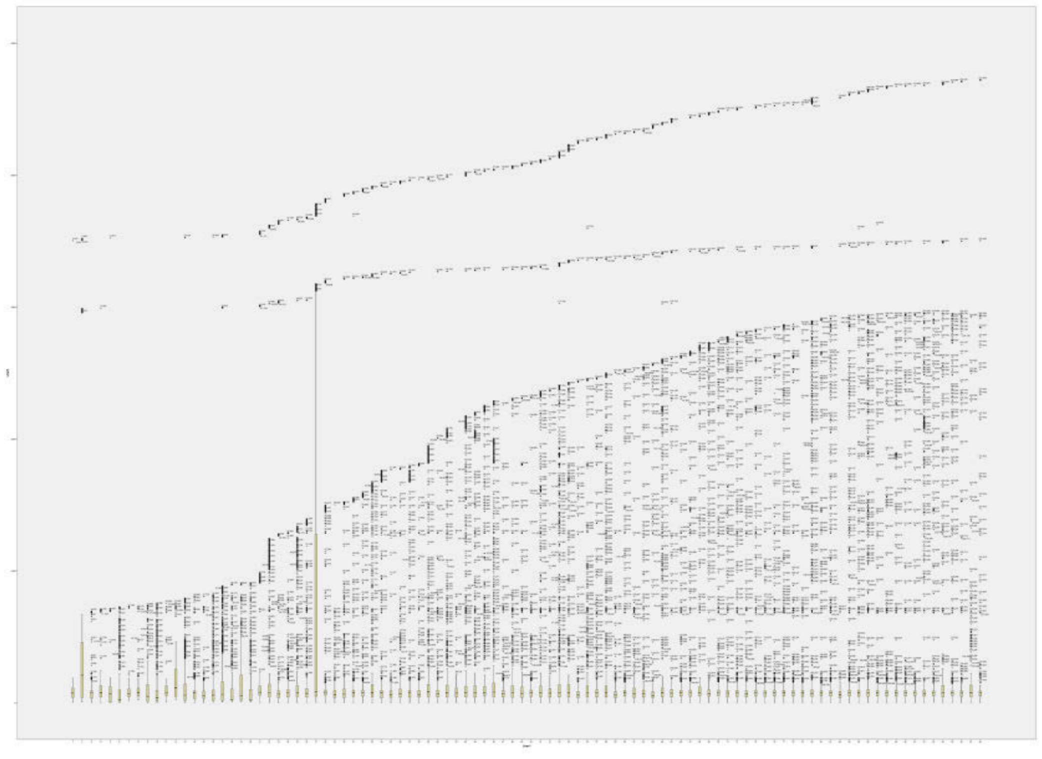


Figure 3.5: Boxplot of User 1 vs User 2 that shows the user gaps between each other.

Node	Frequency (Descending Order)	Position in Sorted Table
3	3250	12
2	3060	21
4	2920	25
5	2711	36
1	2576	46

Table 3.3: Analyse - Descriptive Statistics – Frequencies (High Frequencies repetition and engagements of User1 with other nodes)

Node	Frequency (descending order)	Position in sorted table
88	3543	1
93	3319	2
~		
85	1392	70
39	1367	71
~		
189	1025	89
63	986	90
~		
212	99	185
154	94	186
3481	10	509
172	9	510
~		
3502	2	3253
15	1	3254
~		
4723	1	4518
4724	1	4519

Table 3.4: Analyse - Descriptive Statistics- Frequencies (High Frequencies repetition and engagements of User2) (~ means continuation of numbers)

Table 3.1 and Figure 3.3 demonstrate the high short-time cumulative count with User1 column. Table 3.2 explains the high level of connection time between User1column and User 2. From the dataset, the users those are from column 1 (User 1) got involved in dataset column 2 (User 2) shows a high connection count compare to others.

Table 3.3 shows their position in a table, sorted in descending order. Analysed descriptive statistics frequency of User1 column - in dataset high frequencies repetition and engagements of beginning user of column 1 on by their position in between nodes.

Table 3.4 shows their position in a table, sorted in descending order. Analysed descriptive statistics frequency of User2 column in dataset high frequencies repetition and engagements of beginning user of User2 on by their position in between nodes.

Continuing the Table 3.1, 3.2, and 3.3 outcome research, produced table 3.4 which shows a dramatic change in the frequency of node connections from User 2 to the 4724 users in User2 column of the dataset. The connecting user frequency counting higher to lower, position sorted in the table is significantly 1 to 100 in between frequencies 3543 to 900 (number of connections). And all other nodes in position of frequencies >900 are in significantly positioned of <100 up to 4519 in sorted table. Figure 3.6 below were produced using SPSS and explains the results in Table 3.4.

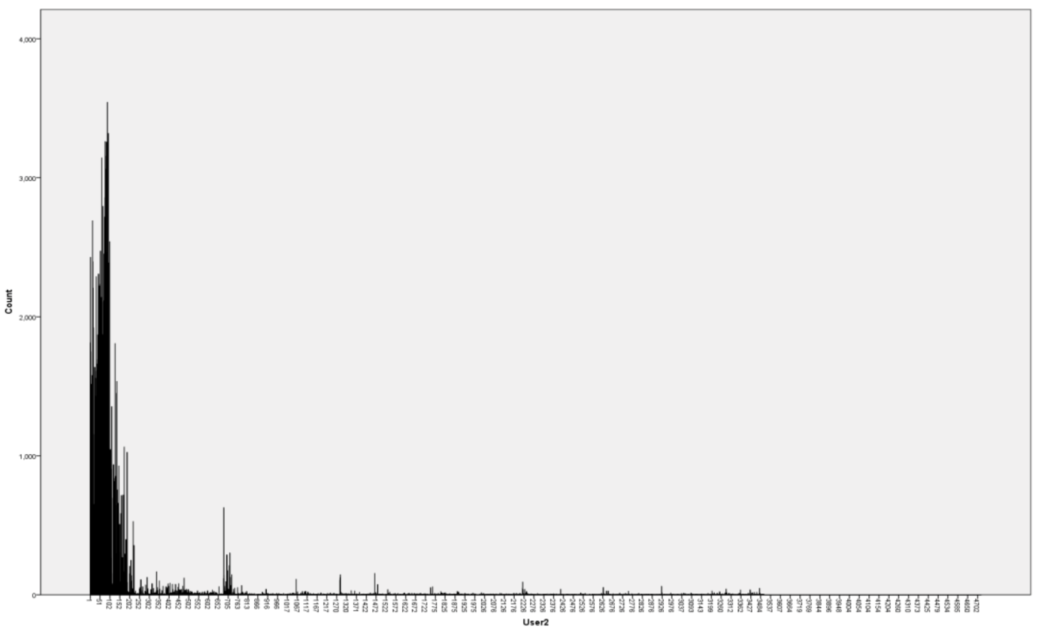


Fig. 3.6 Bar chart of User 2 vs high short-time connectivity

3.3 Conclusion

Content analysis is general set of techniques that are useful for analysing and understanding collections of text. Despite its superior accuracy, content analysis is very time-consuming and often demands an impractical level of manual work. Instead, connection analysis can be used to identify the specific connections between two or more nodes in a database and help to: discover the nodes; understand how they were established; and describe the behaviour between each connection. The results may not be as accurate, but the process of identifying sybil users is much faster, and forms a viable and speedy alternative for administrators with limited resources.

This investigation identified that the Infocom06 dataset does contain users with suspicious activity. A graph-based system was used to arrive at this preliminary outcome. Moreover, connection analysis shows that sybil users can be evaluated and distinguished from honest users with a graph-based system. The graphs demonstrate that sybil activity exists due to significantly higher than average connectivity by some users. For example, User 27 is considered to be highly suspicious as it has the highest connectivity to other users. The connection analysis technique identified the suspicious nodes by producing graphs based on the available connections in dataset demonstrating effectiveness explained in this chapter. The designed algorithm for to generate graphs identified the suspicious node for further research. The experiments showed, the identifying the connections between nodes can be effective to identify sybil node at first step of research before directly experimenting on available content.

Chapter 4
A classified model for sybil detection

4.1 Introduction

A huge amount of data is being collected and stored in databases across the world and its space stations, and this trend continues to increase year upon year. So much valuable knowledge is hidden in these database, it is practically impossible to mine them without an automatic extraction method. Over the years, many algorithms, called nuggets, have been created to extract this knowledge using various methodologies such as classification, association rules, clustering, and many more.

This chapter focusses on classification, and is divided into the following sub-sections:

- Decision trees
- Entropy models
- Information gain
- Random forests
- Gini index-based impurity measures

Section 1: Decision trees are used to split data in to different forms and build a number of trees. Section 2: The C4.5 algorithm is used to create a decision node higher up the tree using the expected value of the classes. Section 3: A random forest is built from the trees.

Classification consists of predicting a certain outcome, based on a given set of inputs. Typically, an algorithm processes a training set, containing a set of attributes and the respective outcome, to discover the relationships between the attributes that make the outcome possible. The algorithm is then given an unseen dataset, called the prediction set, which contains a similar set of attributes without the outcome. The algorithm then analyses the input and attempts to produce a prediction.

Classification models help to predict categorical class labels, which may be discrete or nominal. Constructed models classify data based on a training set and use the resulting class labels values as attributes with which to classify new data.

Decision trees are a very popular method of classification for supervised learning with nominal classes that have dependent labels. Using decision trees to predict nominal class behaviour given a simple ‘yes’ or ‘no’ is straightforward. However in some cases, nominal classes can specify more than two options, and hence which kind of entities responded to which options can be classified and/or predicted.

The first step in the development of this classification model was to evaluate the dataset using a decision tree.

A decision tree was built by first searching for users identified as suspicious. These users were split into the mostly evenly divided groups, given a set of observations and their features. Currently mixed up column 1 or User 1 and column 2 or User 2 connections with other respective columns of starting time, ending time including, etc.

It is worth noting that, a machine learning algorithm has no idea about records or question we asked. The model does not represent a reality but it represent data we gave it to our system and this is why we interested in training and testing.

For example, if model is asked, ‘Which records have a time difference value of 0?’

$$TD = |ST| - |ED|$$

where ST represents the start time, ED represents the end time, and TD represents the result, the model will find all the values where the time difference (TD) is 0. This is not useful when tracking sybil users, so the testing must be supervised. This testing process is integral to learning. Without it there is no measure of a model’s ignorance. In this case we chose the training and testing sets from the available data, rather than from the more commonly used *k-fold* validations.

Choosing samples for training is reminiscent of Goldilocks. If the set is too big, it leads to overfitting where model is not flexible enough to handle real-world data. If the set is too small, it leads to underfitting where model the will not consider enough nuance. And if the set is too skewed, it leads to bias, where the model reflects a reality that does not exist. The training set has to be just right.

To accurately represent a real-world sample, the training data in the first part of the research could be folded in many ways, such as:

- Train on one fold and two folds,
- Test on other *k-1*,
- Rinse and repeat.

If reality changes, the model may need to retrained.

The decision tree in this investigation was split using the features that resulted in the most evenly divided groups – namely their connections. However, this split may not always be symmetrical and even, given the number of user connections will change from node to node, as will the connection time between two nodes.

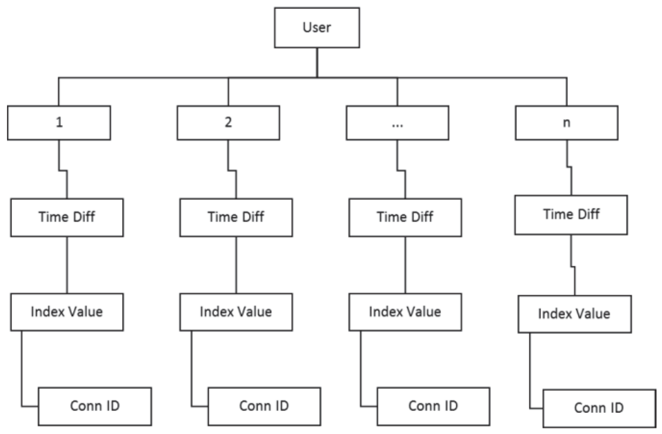


Figure: 4.1 Split of training the dataset

This decision tree helped to classify the model according to previous observations, but dividing the nodes into group presented new questions:

1. Does this user connect to specific user regularly or do they connect randomly to anyone?
2. Is there a chance that this user might connect to each and every available node?

3. If they connect regularly, do they reconnect with friends or to other random nodes?

This decision tree was used in subsequent research of the classification model to predict sybil users and their behaviour.

4.1.1 The C4.5 Algorithm

The decision trees generated by C4.5 can be used for classification and regression. The C4.5 algorithm is developed by Ross Quinlan, which use to generate decision trees. It is an extension of Quinlan's earlier ID3 algorithm. C4.5 often referred to as a statistical classifier.

For example, the training data is a set of

$S = S_1, S_2, S_{3,} \dots$

Each sample S_i consists of a p-dimensional vector

$x_{1,i}, x_2, i, x_{3,i} \dots, x_{p,i}$,

Whereas, x_i represents attribute values or features of the sample, as well as the class into which , S_i falls.

At each node of the tree, C4.5 chooses the data attribute that most effectively splits its set of samples into subsets enriched in one class or the other. The splitting criterion is the normalized information gain (ie Kullback–Leibler divergence) and the difference in entropy (ie information theory). The attribute with the highest normalised information gain is chosen to make the decision. The C4.5 algorithm then recurs on the smaller sub-lists.

This algorithm has a few base cases. All the samples in the list belong to the same class. When this happens, it simply creates a leaf node for the decision tree saying to choose that class. None of the features provide any information gain. In this case, C4.5 creates a decision node higher up the tree using the expected value of the class. Instance of previously-unseen class encountered. Again, C4.5 creates a decision node higher up the tree using the expected value.

- The algorithm operates over a set of training instances, C.

- If all instances in C are in class P, create a node P and stop, otherwise select a feature or attribute F and create a decision node.
- Partition the training instances in C into subsets according to the values of V.
- Apply the algorithm recursively to each of the subsets C.

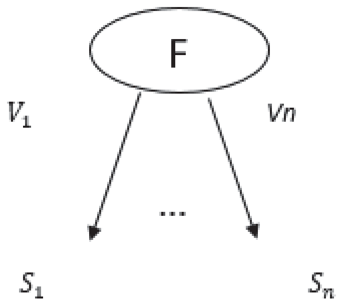


Fig. : 4.2 Decision tree mapping using C4.5 algorithm

Generalised decision trees:

```
Check for basic cases
For each attribute a
Find the normalised information gain ratio from splitting on a
Let a_best be the attribute with the highest normalised
information gain
Create a decision node that splits on a_best
Recur on the sublists obtained by splitting on a_best, and add
those nodes as children of node
```

The author's go on to explain why this approach works, and why it is better for finding sybil users and their behaviour.

Further research idea helps to effectively view the data from many angles. It also eliminates insignificant features and nodes to improve prediction, help build a robust model, and more easily draw parallels with other datasets. However, the current research scope does not consider several crucial details. Each connection set requires an individual model, and observations from honest users and their connection times are irrelevant, so should be discarded. Building model of many nodes at once presents some challenges. We are going to parallelised model building by group it by first column single node in future research.

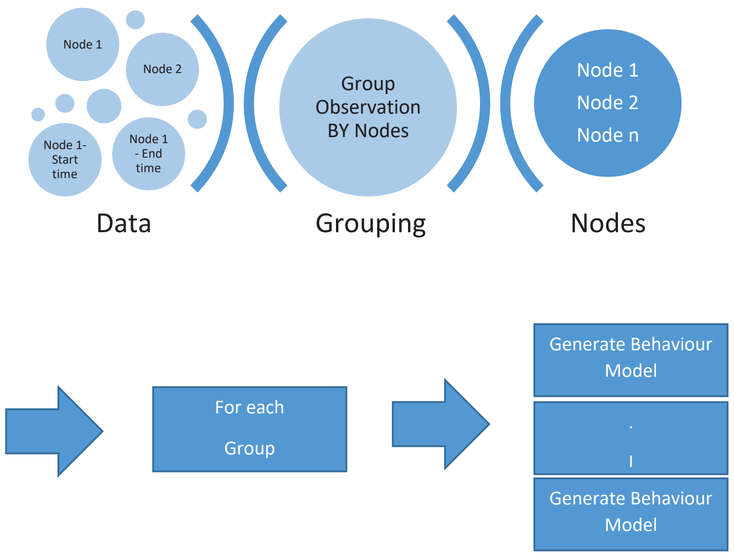


Fig 4.3 Model Diagram

4.2 Decision trees, entropy and gain

In this section, decision trees are used to split data in to different forms and a number of trees are built based on information available in the dataset.

Decision trees work very differently than naïve Bayes. When predicting a sybil user using a decision tree, all the nodes and their possible connections need to be observed. For example, the Infocom06 dataset records 98 users, connecting to over 4000 users multiple times, which generates more than 200,000 connections. The set also includes the length of time the nodes were connected and its index values.

To predict whether or not a user is sybil, each user and their connected node are grouped. This procedure is repeated for each node in User1 column. The Manual intervention is required to determine how many other attempts to predict how many other nodes the selected node is connected to. Other factors for consideration include: whether the selected node is connected to a limited or large number of other nodes; whether *those* nodes have further connections that are limited or large; and whether the selected node is connected to other nodes for a specific or non-specific time.

Trees were selected for further random forest processing, based on these determinations. The key to generating a decision tree for each node is to glean what type of connections the node has made and why it made those connections. Examining each one of these attributes, such as User1 column and User2 column node, start and end time of connection, index values, etc. and try to meet the best interest possible about assuming its connection, whether the selected node is suspicious and what other factors may influence the consideration of a node as sybil.

ATTRIBUTE	POSSIBLE VALUES
User 1	Continuous
User 2	Continuous
Starting Time	Continuous
Ending Time	Continuous
Index Value	Continuous
Connection ID	Continuous
Time Difference	Yes, No
Index Value one	Yes No
Connection ID Zero	Yes No

Generally speaking, attributes in the available data are examined and then used to split the data into subsets. For example, the number of connections a given node has made could be: specific (ie regular/honest); limited (ie only few connections); or large (ie connect to most of available or more than selected average). Additionally, those three subsets will contain further subsets of information.

If subset is pure then it is honest. That means if the node had specific connections, and connected over an unspecified time, the process will stop. Otherwise it will continue to investigate the behaviour and try to further split the data – a sort of variation on a divide and conquer algorithm.

The new datasets were tested for predictions next, to see which examples fell into which subsets. The dominant class is then used for that subset. For example, if the selected node always connects to the average number of nodes for an unspecified time, then is it certain that the selected node is honest and the branch is split at higher level displaying remaining values.

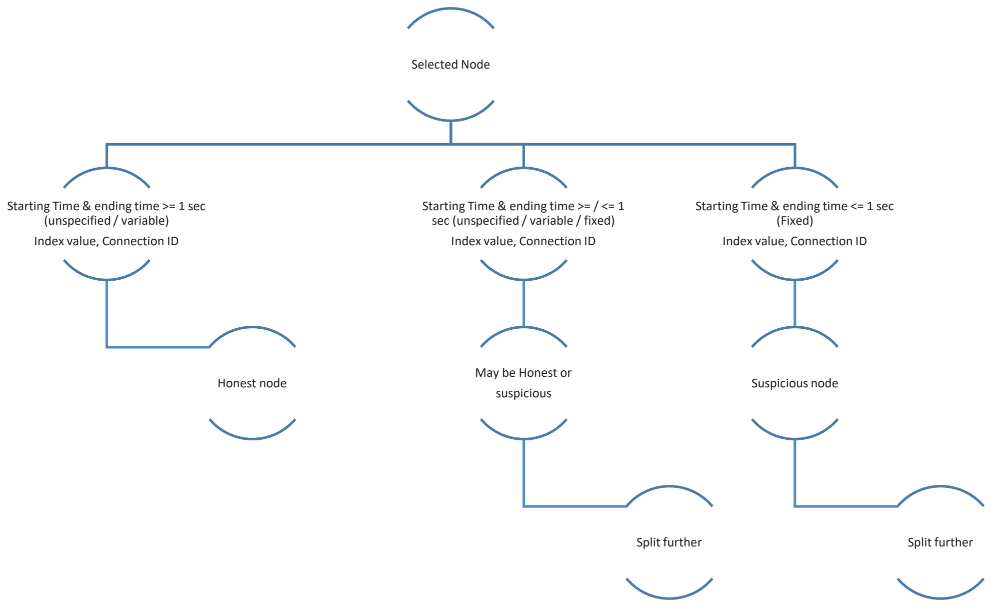

Selected Node
Starting Time & ending time >= 1 sec (unspecified / variable)
Index value, Connection ID
Starting Time & ending time >= / <= 1 sec (unspecified / variable / fixed)
Index value, Connection ID
Starting Time & ending time <= 1 sec (Fixed)
Index value, Connection ID
Honest node
May be Honest or suspicious
Suspicious node
Split further
Split further

Figure 4.4 Generalised decision tree map for Infocom06

If the selected node is categorised as suspicious or considerably suspicious, the node is divided into even more branches based on the start and end time of its connections. These branches of the decision tree:

- have a lower connected node count; and/or
- are less than or equal to the time difference

The tree will split further again if *both* values are true, otherwise the node will be deemed honest.

The algorithm stops when two pure subsets are found, because there is no need for further division after another subset is deemed pure, and there is no need to further investigate sybil activity. The next node is selected and the tree building process is restarted. The node is considered to be honest if there is a high probability of a low connection count, and the node has connected with other nodes for an undefined specific time. If a node in the second branch of the tree is not pure, the algorithm checks for index value and will repeat the procedure. If the second branch is also not found to be pure, then based on column six of Infocom06 (connection id) value of 0's count (>= avg 0's count), it will further branch the tree and the node will certainly be considered sybil.

Finding the split:

Number of branches *fi* – each feature in dataset

Let *fi* be the feature with the greatest gain

Create a decision node that splits on *fi*

For each split *Spl* on *fi, FindSplit(Spl)*

```
C4.5 algorithm:
Split (node, (number of branch (fi)))
A <- the best attributes for splitting the fi
Decision attributes for this node <- A
For each value of A, create new child node
Split training fi to child nodes
Do each child node / subset
If subset is Pure: STOP
Else
Split (child_node, (subset of fi)
Consider node: Considerably suspicious / honest
Repeat
If subset is Pure: STOP
Consider node: Honest
Consider node: Suspicious
Repeat
If subset is Pure: STOP
Else
Consider node: Sybil and user for RF (Random forest)
```

Selecting the best attribute: Nodes can be connected based on: time of connection; index value; or connection id's 0's count. There are pros and cons associated with each attribute. The *purity* of splits have to be measured, so the ideal choice is the attribute with all pure subsets to help reduce the dataset and eliminate data for further examination.

Some cases have a 50% chance of being pure or impure. So split the generated subset in pure side is good and not to split the generated subset likewise have complete impurity. A matrix that can measure both the purity of the subset and its uncertainty is required. The uncertainty value is a measure of probability that, after the data of a particular subset is split, a random item within that subset is positive or negative. - A completely uncertain number, with a 50/50 chance of being positive or negative, demonstrates that the node is honest or suspicious.

We cannot use a posterior probability – the probability that an observation will fall into a group before the data is collected – because the subset needs to be symmetrical. It means, a pure subset or honest node has low regular connection with uncertain frequency of time connections which is good similarly pure subset or honest node have high connection with uncertain frequency of time connections. So it can't be a probability of positive. It must be some that is symmetric of positive side and negative side.

4.2.1 Entropy

Calculating entropy is a way to measure the uncertainty of a class in a subset of *fi*. Entropy is defined as:

$H(S) = -p(+) \log_2 p(+) - p(-) \log_2 p(-)$ *bits*

- *S* is subset of training example
- *p(+) / p(-) ... % of positive / negative examples in S*

Interpretation is needed in this context. Assuming item X belongs to S, how many bits are needed to tell if X is positive or negative?

If the subset has, for example, 10 positive and 10 negative samples, then 1 bit is spent. In the cases that are less certain than the pure set, but more certain than a coin flip, entropy gives a number between 0 and 1 which represents a fractional number of bits – except it is a fractional number of bits which is little strange.

Entropy calculations are based on binary values of yes and no, or 1's and 0's and. The original dataset did not have any text values.

Hence, if the impure subset = 1 bit:

H(S) = - (number of purity / total number) log2 (number of purity / total number) - (number of impurity / total number) log2 (number of impurity / total number) = 1 bit

If the subset is pure it = 0 bits:

H(S) = - (number of purity / total number) log2 (number of purity / total number) - (number of impurity / total number) log2 (number of impurity / total number) = 0 bits

Entropy tells us how pure and impure is one set and subset.

Now the information must be segregated from several different subsets, because the attribute selected for the split has different values. A not-so-simple average is used. The aim is to have as many items considered to be honest as possible in pure subset, and a drop in entropy after the split is expected:

$$Gain\ (S, A) = H(S) - \sum_{V \in value(A)} \frac{|Sv|}{|S|}\ H(Sv)$$

This is taken in to account when adding the entropy value, by putting a weight on each entropy. A weight is put on each subset, which is the size of that subset divided by the overall number of *fi* there are at this split node.

V is a possible or particular value of A

S is a set of fi (example) {X}

Sv subset where X_A= V ,which is all the fixed time or all the maximum connections or all the connection id's 0 count.

This is the entropy of those such subsets and the weight indicates what proportion of the items failed in to the rather fixed time that is no time difference or all the maximum connections that is highest number of connectivity. So items failed in information gain calculation to fixed time is multiplied by the how pure was the fixed time or all the maximum connections.

If the resulting subset is large and pure then it’s good and If the resulting subset is small and impure then it’s bad. If the result is good if it returns a large pure subset, any sized then it is good or if we get small or big impure subset it is bad.

In summary, the average purity is weighted by the size of the set’s average purity after the split on attribute A, because there were some positive and some negative splits.

Looking at, the difference in entropy before and after the split is the determining point at which interpret whether we are sure how much we are certain before split and how much more certain after the split. That is which node is considered as honest or which going to be consider as, suspicious. This is called information gain.

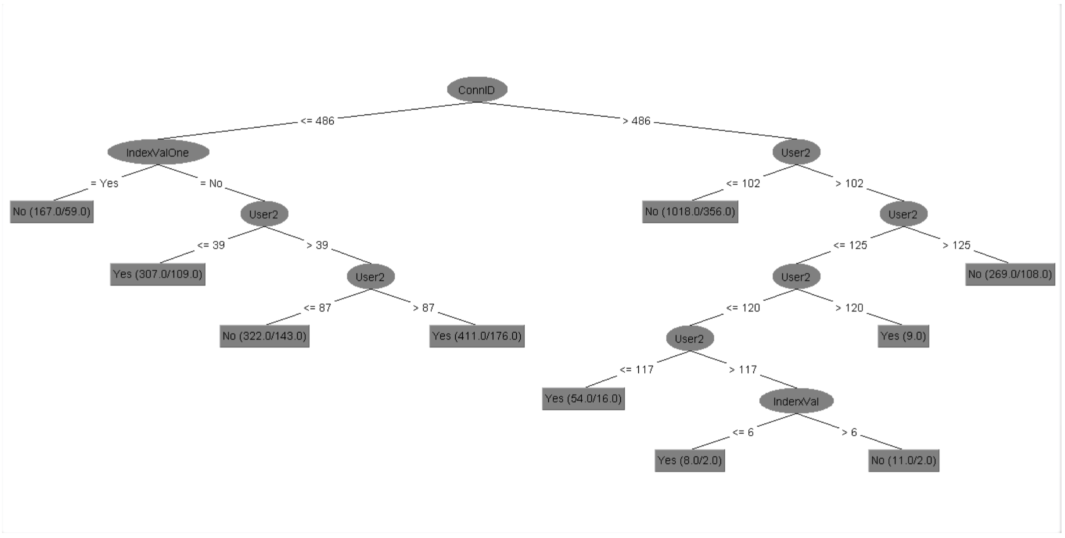


Fig. 4.5 Decision tree - node1 generated in WEKA

4.2.2 C4.5 processing

The part of the research contributes to classification and regression models using the C4.5 algorithm for outcome generation. To prepare, the dataset was split based on the User1. There are 98 users in the first column, each connecting to many other users in the second column (User2). The split carries information in both columns.

4.2.2.1 The issues

The computer suffered memory full errors, when tree plotting started. After running algorithm for a while, it stop working the programming applications at first step. This dataset contains over 200,000 records of multiple user connections, and even avoiding special functions did not prevent runtime memory problems.

Additionally, even if the tree had been plotted with a highly configured computer and processing system, the results may not be have clearly justified and predicted the outcome. It may have plotted a very large tree which would be impossible to interpret, for example.

4.2.2.2 Overcoming the issues

To avoid later confusion, the dataset was divided into parts. The best division was based on User1 which provided a clear track to better results for sybil prediction. After splitting dataset, without damaging the raw data, which brought in pieces we were able to use defined algorithm in this research.

Ultimately, overcoming these obstacles resulted in reprogramming the algorithm in a better way. Manual work was increased, but the processing time decreased and the solutions yielded better results.

For the purpose of flexibility in the entropy calculation, three new columns based on time difference (start time and end time), index value () and connection id () were added. The time difference is the start and end time of each connection. As per the observations, some user had the same start and end time, which is a time difference of 0. Hence, column 7 named TimeDiff was added to store a binary value of yes or no, with yes representing a time difference greater than zero. The next column stored a similar binary value for IndexValOne, where yes indicates that index value has value one in front of the connection. The third and final column holds the binary values for connection id, with yes indicating that column 6 has a value of zero in front of the connection.

4.2.2.3 The splitting process

To split the dataset an automated function in R was used to build the data frame. The complete original dataset was saved in to one frame, and a new variable for data frame was created.

A specific number of users was selected from column one of the dataset. Data with all columns related to the specific user number was fetched and exported to the group of data into new files and saved with a user number for future recognition. This could be accomplished manually or with a loop. Doing this manually provided an opportunity to observe the file creation process for safety and future processes.

4.2.3 Binary count of columns

4.2.3.1 Total count for each node and connection

After the split process in number of datasets, the total number of specific dataset user connections to other users was calculated. For example, calculating the entropy for the first level 1(split), requires knowledge of how many times node 1 connected to the second column of node 3. This brings us the total count of node 1 and 3 connection.

The inbuilt R function and library "plyr" was used for re-processing which has tools for splitting, applying and combining data. The process is as follows: fetch the split dataset for a specific user in one frame; create a new variable for the data frame; use the inbuilt count function to access User1 and User 2 columns of the dataset; return the count of each pair of connections; fetch data from User1 and User2 related to the specific user number selected with its count; export the group of data into a new file, and save the pair count value for future recognition; with its connection frequency between User 1 and User 2.

Again, this could be done either with a loop or manually. Taking the manual approach provided opportunities to observe and ensure the integrity of the data was maintained for the algorithm. The process for each dataset split for each node of User1 column is the same.

4.2.3.2 Time difference count for each node and connection

The entropy calculation, that takes place at the second stage which include one more column compare to previous stage (section 4.2.3.1), requires a total count of TimeDiff's yes/no values for each pair and, as previously, the inbuilt R function was used as follows: fetch the split dataset for a specific user in one frame; create a new variable for the data frame; use the inbuilt count function and access the User1 and User2 columns of the dataset in addition to the TimeDiff column; return the TimeDiff count for each pair; fetch the data from User1, User2 and TimeDiff of the specific user number selected with its count; export this data into a new file, saving the pair count for future recognition; saving User 1, User 2, Time Diff (Yes/No) their frequency of each Yes and No for that specific pair. Again, this step was processed manually for observation, and a cross-check that the count of pair 1–3–Yes plus a count of 1–3–No was equal to the total frequency count of Table 4.1 was made.

4.2.3.3 IndexValOne count for each node and connection

The entropy calculation at the next split requires a count of the yes/no values in the IndexValOne column. The same process, and the same inbuilt R function, as the TimeDiff count was followed on all datasets from the first split.

Fetch the split dataset for the specific user in one frame; create a new variable for the data frame; use the inbuilt count function to access the first three columns of the dataset in addition to the TimeDiff and IndexValOne columns; return the count for each pair of connections; fetch the data from that specific user number selected with its count User1 User2, TimeDiff and IndexValOne; export that data into a new file, saving the pair count for future recognition; saving User 1, User 2, TimeDiff and IndexValOne, and the frequency of each yes/no for that specific pair. The best way to process this step is, again, manually, but larger datasets could use a loop.

4.2.3.4 ConnIDZero count for each node and connection

A similar calculation of the frequency count for the ConnIDZero column in addition to IndexValOne is required, which will return mostly 0 entropy values, with the exception of nodes with irregular patterns that were observed in dataset.

This we have not used much but we still need this to predict one of the sybil node which explain further in research. The process follows the same basic set of steps: fetch the split dataset for specific user in one frame; create a new variable for the data frame; use the inbuilt count function to access the first three columns of the dataset in addition to plus the TimeDiff and IndexValOne columns; return the counts for each pair of connections; fetch the data from column 1, column 2, TimeDiff, IndexValOne of the specific user number selected with its count; export the group of data into new files, saving pair count number for future recognition; save User1, User2, TimeDiff, IndexValOne, and ConnIDZero and their frequency of each yes and no for that specific pair.

4.2.4 Entropy generation and analysis

4.2.4.1 Analysis of user vs TimeDiff

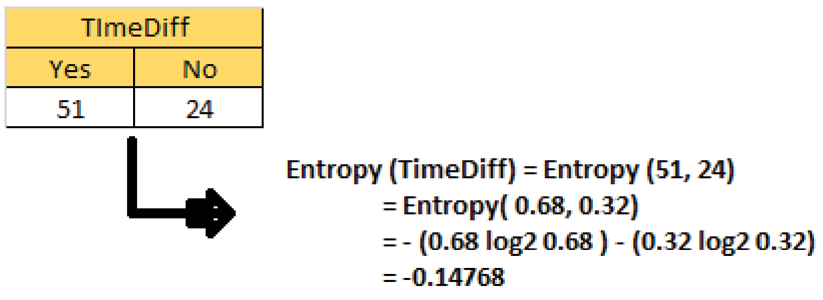
Processing the entropy for each node requires all the datasets created in the previous steps above. The entropy model stated above in research for each node of the Total count for each node and connection datasets that we have generated in 4.2.3.1 section and Time Difference count for each node and connection 4.2.3.2 section.

Processing this entropy requires two saved datasets for each node – the dataset that contains User1, User2 and frequency, and the dataset that contains User1, User2, TimeDiff and frequency.

The below example shows entropy generation for the first node, based on user count and time difference, and is calculated with the help of User1 and User2 pair and other similar node total in dataset with TimeDiff and its frequency.
then further calculate entropy for both side nodes

$$E(S) = \sum_{i=1}^{c} -p_i \log_2 p_i$$

User		
User1	User 2	Total Connections
51	24	75



TImeDiff	
Yes	No
51	24

Entropy (TimeDiff) = Entropy (51, 24)
= Entropy(0.68, 0.32)
= - (0.68 log2 0.68) - (0.32 log2 0.32)
= -0.14768

Figure 4.6: Example of a one-attribute entropy calculation

Entropy using the frequency table of one attribute:

- Selected node = 1 (column 1, first node)
- Total node connections to the column's node is 75.
- Time = 0 seconds - 24 nodes are connected
- Undefined time >= 1 second are 51 nodes

$$Entropy\ H(S) = -\frac{51}{75} log2 \frac{51}{75} - \frac{24}{75}\ log2 \frac{24}{75}$$

$$= -\ 0.147686504$$

As specified, the entropy model has a right side and a left side. The left side tends toward 'yes' or calculation of leave node entropy (to 0) which helps to eliminate the complications in the end results and predictions. The right side tends toward 'no' or leave entropy to non-zero value(an entropy value >0). In the above example, 75 nodes were divided – also called a root node for 1-3 pair decision tree – and generated a small tree in which 51 nodes returned yes and tended to the left.

Entropy uses the frequency tables of two attributes.
To calculate the further leaf frequencies we used the dataset of node 1 we created in Time Difference count for each node and connection 4.2.3.2 generated in section including IndexValOne count for each node and connection 4.2.3.3 generated in section.
The following entropy calculated based on the first frequency we calculated with the help of User1, User2, TimeDiff, frequency pair and other similar node dataset with IndexValOne and its frequency.

In this example, 51 nodes had no further leaves with which to calculate entropy so the output on the left side was zero.

$$Entropy\ H(S_{Honest}) = \ 0$$

On the right side, 24 nodes comes in to group of frequency, in which one node has an IndexValOne and the other 23 nodes have values that fall into other categories.

$$Entropy\ H(S_{Suspicious}) = -\frac{1}{24} log2 \frac{1}{24} - \frac{23}{24}\ log2 \frac{23}{24}$$

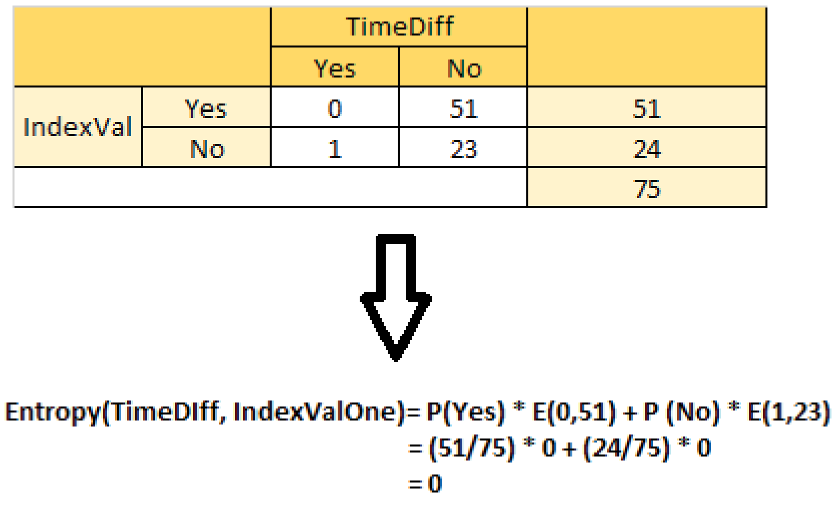
$$= 0.132197916$$

In this way, all the entropy values, for each pair of each datasets, for all the above split datasets and their frequencies, are calculated.

```
Algorithm: Entropy generation per user - TimeDiff datasets
1  Infocom06tbUsers <- Read dataset – Total user frequency count
   and save in data frame
2  N1 =Calculate length of dataset
3  Infocom06tbTimeDiff <- Read dataset – user frequency count
   based on TimeDiff and save in data frame
4  N2 =Calculate length of dataset
5  Data1 = Null #empty data frame to save new data
6  Loop L1 Infocom06tbUsers  until N1 count
7  Access and save each row and column element in to variables
8  Loop L2 Infocom06tbTimeDiff until N2 count
9  Access and save each row and column
10 Check L2 column 2 user with L1 column 2 user are equal
11 Check L2 Column 2 user with next L2 Column 2 user are equal
12 Check L2 Column 2 user same as next L2 Column 2 user then
13 Calculate Entropy
14 Else Entropy set to 0
15 Bind Data in Data1 frame
16 Repeat all from same pairs in Infocom06tbUsers
17 Save and write Data 1 in NewFile
```

Repeat this algorithm for all users and time difference frequency count datasets. This generates new files for the pair with User1, User2, Frequency 1, and EntropyTimeDiff.

The above dataset worked perfectly, calculating the entropy for all values in most of the datasets with only one issue, as outlined below.



		TimeDiff		
		Yes	No	
IndexVal	Yes	0	51	51
	No	1	23	24
				75

Entropy(TimeDIff, IndexValOne)= P(Yes) * E(0,51) + P (No) * E(1,23)
= (51/75) * 0 + (24/75) * 0
= 0

Fig. 4.7: sample of two or more attribute entropy calculation

4.2.4.2 First row mismatch issue in dataset

As per the above algorithm, the entropy successfully calculated have similar node in second loop. For example, Check L2 User2 users with next L2 User2 users are successful and dataset process the outcome perfectly.

In some split datasets, the above checked condition gets false at start, because first row value for this condition does not match and it returns the all NA values for each entropy. It happens because there are no corresponding binary values for those attributes in the other dataset, and hence there is no frequency for the algorithm to further process.

As shown in Tables 4.6 and 4.7, a binary One (Yes) value count for Users user pairs 7 and 18 does not exist.

To overcome this problem, the else part of the condition, where R automatically manages the loop for such datasets, was eliminated. Surprisingly, this problem occurred in datasets that had a first row mismatch.

```
Algorithm: Entropy generation per user - TimeDiff datasets - to
handle mismatch first row values in both datasets
1   Infocom06tbUsers <- Read dataset – Total user frequency count
    and save in data frame
2   N1 =Calculate length of dataset
3   Infocom06tbTimeDiff <- Read dataset – user frequency count
    based on TimeDiff and save in data frame
4   N2 =Calculate length of dataset
5   Data1 = Null #empty data frame to save new data
6   Loop L1 Infocom06tbUsers  until N1 count
7   Access and save each row and column element in to variables
```

```
8  Loop L2 Infocom06tbTimeDiff until N2 count
9  Access and save each row and column
10 Check L2 column 2 user with L1 column 2 user are equal
11 Check L2 Column 2 user with next L2 Column 2 user are equal
12 Check L2 Column 2 user same as next L2 Column 2 user then
13 Calculate Entropy
14 Bind Data in Data1 frame
15 Repeat all from same pairs in Infocom06tbUsers
16 Save and write Data 1 in NewFile
```

4.2.4.3 Last row mismatch issue in dataset

Similarly, the algorithm was not able to trace a last row mismatch because almost all datasets have a single value at the end of their user connections (Figure 3.6. 6 Bar diagram for User 2 vs high short-time connectivity).

Every final pair, in almost all datasets, made only one connection with each other, which fell under an either yes or no binary value.

To solve this issue, entropy was calculated manually and all tables were updated. It is worth noting that the entropy value for all was 0, because there is no other binary element present for comparison.

4.2.4.4 Analysis of TimeDiff vs IndexValOne

Next, the entropy for time difference and its frequency, and IndexValOne and its frequency tables was calculated. This requires a cross-check with more conditions from the table listing time difference and its frequency.

```
Algorithm: Entropy generation TimeDiff - IndexValOne datasets
1  Infocom06tbTimeDiff <- Read dataset – TimeDiff frequency count
   table data and save in data frame
2  N2 =Calculate length of dataset
3  Infocom06tbUsersIndexVal <- Read dataset – IndexValOne
   frequency table data and save in data frame
4  N3 =Calculate length of dataset
5  Data2 = Null #empty data frame to save new data
6  Loop L2 Infocom06tbTimeDiff until N2 count
7  Access and save each row and column element in to variables
8  Loop L2 Infocom06tbUsersIndexVal until N3 count
9  Access and save each row and column
10 Check L2 column 2 user with L3 column 3 user are equal
11 Check L3 Column 2 user with next L3 Column 2 user are equal
12 Check L3 TimeDiff user with next L3 TimeDiff are equal
13 Check L3 IndexValOne with next L3 IndexValOne are not equal
14 Check L2 TimeDiff with L3 TimeDiff are not equal
```

```
15 Check L3 Column 2 user same as next L3 Column 2 user then
16 Calculate Entropy
17 Else Entropy set to 0
18 Bind Data in Data1 frame
19 Repeat all from same pairs in Infocom06tbTimeDiff
20 Save and write Data 2 in NewFile
```

The above dataset also worked perfectly, calculating the entropy values for all datasets, without any issues or mismatches except for last row issue identified previously. The same solution was applied: entropy was calculated manually, again with a value of 0 for all, and all tables were updated.

4.2.4.5 Analysis of IndexValOne vs ConnID

Calculating the entropy model for IndexValOne and its frequency, and ConnID zero and its frequency tables, requires a further cross-check against more conditions from the time difference, index value and connection ID tables.

```
Algorithm: Entropy generation IndexValOne - ConnIDZero Datasets
1  Infocom06tbUsersIndexVal <- Read dataset – IndexValOne
   frequency count table data and save in data frame
2  N3 =Calculate length of dataset
3  Infocom06tbUsersConnID <- Read dataset – ConnIDZero frequency
   count table data and save in data frame
4  N4 =Calculate length of dataset
5  Data3 = Null #empty data frame to save new data
6  Loop L3 Infocom06tbUsersIndexVal until N3 count
7  Access and save each row and column element in to variables
8  Loop L4 Infocom06tbUsersConnID until N4 count
9  Access and save each row and column
10 Check L3 column 2 user with L4 column 2 user are equal
11 Check L4 Column 2 user with next L4 Column 2 user are equal
12 Check L4 TimeDiff user with next L4 TimeDiff are equal
13 Check L4 IndexValOne with next L4 IndexValOne are equal
14 Check L4 ConneIDZero with next L4 ConneIDZero are not equal
15 Check L4 Frequency with L3 Frequency are not equal AND
      (L3 Frequency is equal to (L4 Frequency + Next L4 Frequency)
16 Check L3 Column 2 user same as next L3 Column 2 user then
17 Calculate Entropy
18 Else Entropy set to 0
19 Bind Data in Data1 frame
20 Repeat all from same pairs in Infocom06tbUsersIndexVal
21 Save and write Data 3 list in NewFile
```

Thus, the entropy calculations for all the nodes and connections in the generated datasets are complete. The outcome of the last table shows a 0 entropy value for every node in each table except for one – number 55. This oddity is discussed in the following section.

4.2.5 Information gain analysis and generation

Using the above algorithms and model, entropy was generated up to the third split, and revealed an unexpected outcome with User 55. A cross-check of the relationship between user 55 in User1 column and other nodes provided further explanation.

The process gain calculation, with the help of entropy, only uses an entropy calculation based on time difference, because there is no need to calculate gain for second and third stage of entropy when the first stage result is 0. It happened because of entropy single binary value of 0.

Hence, the information gain model for the time difference tables is:

$$Gain\ (S, A) = H(S) - \sum_{V \in value(A)} \frac{|Sv|}{|S|}\ H(Sv)$$

Where,
S – user TimeDiff total entropy
A – attribute selected for .
So the Information Gain between attributes A and clad label of S continuing the first example with the gain model:

$$\begin{aligned}&\text{Gain (S, Time Difference)}\\ &\qquad = (-0.147686504) - \frac{51}{75} * (0) - \frac{24}{75} * (0.132197916)\\ &\qquad = -0.10538317088\end{aligned}$$

```
Algorithm: Calculate information gain for each node and dataset
1  Infocom06tbUsersIG <- Read dataset - Total user frequency count
   and save in data frame
2  I1 =Calculate length of dataset
3  Infocom06tbTimeDiffIG <- Read dataset - user frequency count
   based on TimeDiff and save in data frame
4  I2 =Calculate length of dataset
5  IGaintbUser = Null #empty data frame to save new data
6  Loop G1 Infocom06tbUsersIG until I1 count
7  Access and save each row and column element in to variables
```

```
8  Loop G2 Infocom06tbTimeDiffIG until I2 count
9  Access and save each row and column
10 Check G2 column 2 user with Last G2 column 2 user are not equal
11 Check G2 column 2 user with G1 column 2 user are equal
12 Check G2 column 2 user with next G2 column 2 user are equal
13 Calculate Gain
14 Bind Data in Data1 frame
15 Repeat all from same pairs in Infocom06tbUsersIG
16 Save and write IGaintbUser list in NewFile
```

Some of the returned gain values were either 0 or NA because it could not find second value and equation generate 0 results.

Based on the calculated information gain for each table and each node, the single maximum gain and their node number is calculated to help predict sybil users.

4.2.6 maxGain analysis

As per the research flow, the next stage is to analyse maxGain.

maxGain is the maximum calculated value of the information gain of a single node among other nodes. maxGain, in this research was calculated based on User1 users or split datasets that we have generated to calculate the information gain for each node. The max function of R was used capture the max value of a single dataset including its other row values.
The results were saved in a new file, one by one, as the maxGain for each information gain dataset was calculated.

```
Algorithm: Calculate maxGain
1  Infocom06tbUsersIG <- Read dataset - Total user frequency count
   and save in data frame
2  M1 =Calculate length of dataset
3  IGainMaxtbUser = Null #empty data frame to save new data
4  maxGain is max value of Infocom06tbUsersIG Gain column
5  Loop Gn Infocom06tbUsersIG from 1 to M1 count
6  Access and save each row and column element in to variables
7  Check Gn  Gain with maxGain are equal then
8  Set other values of that row to variables
9  Bind Data in IGainMaxtbUser frame
10 Repeat all from same pairs in Infocom06tbUsersIG
11 Save and write IGainMaxtbUser row to list in File
```

The algorithm's resulting maxGain calculations for each node are shown in Table 4.12.

4.3 Random forests and the Gini index

Random forest is an ensemble-style, learning-based classification and regression technique, commonly used in predictive modeling and machine learning. In regular decision trees, one decision tree is built and a random forest algorithm determines the number of trees that are built during processing. A vote from each of the decision trees decides the final class of a case or object. This is called an ensemble process, also known as a democratic process, where many decision trees are built and used in the process of random forest algorithms.

Making a random forest from trees follows the procedure below:

1. Determine the number of trees to build (n)
2. For each tree
   -Take a training set that include only some of column
   -Build a decision tree and prune it.
3. Traverse all n trees:
   -At each level, find the mode split (Mode averaging to find best possible tree)
      - Make this split at this level of the 'best tree'
4. The best tree, at the end, is used for classification

Data frames have two dimensions, observations (rows) and variables (columns). To build a decision tree, samples of a data frame are selected specific node, along with the selection of a subset of variables for each of the decision trees. Both the sampling of the data frame and the selection of the subset of variables are done randomly

There are two key advantages for using random forests in analytics. The chances of over-fitting are reduced, and they provide higher model performance and accuracy.

Random forests use Gini index-based impurity measures to build decision trees. Gini indexes are also used to build classification and regression trees (CART).

The Infocom06 dataset has 9 independent input variables and a target variable. The target variables TimeDiff, IndexValOne and ConnIDZero are binary.

The first split of the User 1 dataset resulted in 45% of connection to other users with a time difference, and 55% with no time difference.

Accordingly, the development and validation samples that must now be created should carry a similar distribution of the target variable If the target variable is either a response variable or a class of target, a decision tree will be built.. The current data frame has a list of independent variables, which can be turned into a formula and then passed as a parameter to the random forest.

With the sample data selected and formula for building a random forest model established, R code is used to build a random forest of 500 decision trees.

R Code to build a random forest

```
names(Infocom06)
#The target variable
table(Infocom06_DC$TimeDiff)/nrow(Infocom06_DC)
#Now, we will split the data sample into
development and validation samples
sample.ind <- sample(2,
        nrow(Infocom06_DC),
        replace = T,
        prob = c(0.6,0.4))

check.user.dev <- Infocom06_DC[sample.ind==1,]
check.user.val <- Infocom06_DC[sample.ind==2,]

table(check.user.dev$TimeDiff)/nrow(check.user.dev)
table(check.user.val$TimeDiff)/nrow(check.user.val)

class(check.user.dev$TimeDiff)
```

```
varNames <- names(check.user.dev)
# Exclude ID or Response variable
varNames <- varNames[!varNames %in% c("TimeDiff")]

# add + sign between exploratory variables
varNames1 <- paste(varNames, collapse = "+")

# Add response variable and convert to a formula object
rf.form <- as.formula(paste("TimeDiff", varNames1, sep
= " ~ "))
check.user.rf <- randomForest(rf.form,
             check.user.dev,
             ntree=500,
             importance=T)
plot(check.user.rf)
```

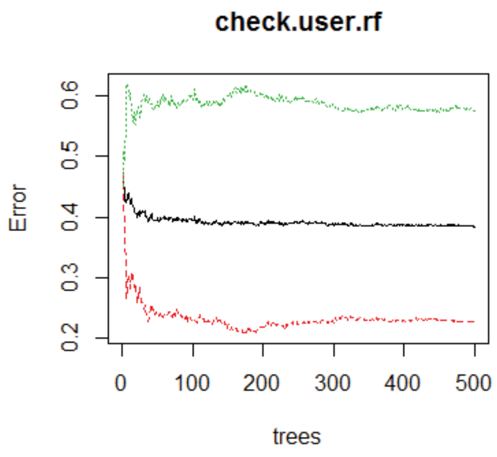


Figure 4.8: Error rate across decision trees

The random forest algorithm produced a forest made up of 500 decision trees. Plotting the error rates across each tree indicated that there was no a significant reduction in error rate after 100 trees.

4.3.1 Variable importance

Variable importance plots are also a useful processing tool, and are plotted using the varImpPlot function. The top five variables were selected and plotted to show the model's accuracy and Gini value. A table showing the decreasing order of importance based on a measure (1 for model accuracy and 2 node impurity) was also created. Based on random forest variable importance, the variables could be selected for any other predictive modelling techniques or machine learning(Table 4.1).

MeanDecreaseGini	Variables
82.4	ConnID
70.2	ConnEndTime
68.2	ConnStartTime
62.4	User2
55.4	InderxVal
1.8	IndexValOne
1.8	ConnIDZero
0	User1

Table 4.1: Gini index calculation

4.3.2 Confusion matrix

A confusion matrix and its statistics was calculated from the random forest and pruning the decision trees.

R Code for confusion matrix

```
## Loading required package: lattice
## Loading required package: ggplot2
# Create Confusion Matrix
confusionMatrix(data=check.user.dev$predicted.response,
                reference=check.user.dev$TimeDiff,
                positive='Yes')
```

Prediction	No	Yes
No	775	259
Yes	81	438

Table 4.2: Confusion Matrix

Accuracy	0.781
95% CI	(0.76, 0.801)
No Information Rate	0.551
P-Value [Acc > NIR]	<2e-16

Table 4.3: Confusion Matrix Statistics

The matrix showed a very commendable accuracy of 78.10 %.

Predicting a response for the validation sample and calculating the model's accuracy on the sample follow as the next steps.

```
# Predicting response variable
check.user.val$predicted.response<-predict(check.user.rf
,check.user.val)

# Create Confusion Matrix
confusionMatrix(data=check.user.val$predicted.response,
                reference=check.user.val$TimeDiff,
                positive='Yes')
```

Prediction	No	Yes
No	431	284
Yes	135	173

Accuracy	0.59
95% CI	(0.56, 0.621)
No Information Rate	0.553
P-Value [Acc > NIR]	0.00903

Table 4.4: Confusion Matrix and Statistics for response variable

The accuracy level dropped to 59%, but is still significantly higher than existing research .

4.3.3 Decision tree pruning

```
R Code to code to prune a decision tree
```

```
install.packages("rpart")

install.packages("randomForest")
install.packages("pmml")

library(rpart)

library(randomForest)

library(pmml)

library(parallel)

library(foreach)

library(doParallel)


Infocom06_sample <-
read.csv("C:/Users/Anand/Desktop/Project1/RProgTree/inf
ocom06_Dummy.csv")

dec_tree <- rpart(TimeDiff ~ User1 + User2 +
ConnStartTime + ConnEndTime + InderxVal + ConnID
+IndexValOne+ConnIDZero, method="class",
data=Infocom06_sample, cp=0.0001)

prune(dec_tree, cp=0.006)

plotcp(dec_tree)

plot(dec_tree,main = "Sybil")

text(dec_tree, use.n = TRUE, cex = 0.5)

cl <- makeCluster(detectCores())

registerDoParallel(cl)

rf <- foreach(ntree =rep(5,detectCores()), .combine =
combine, .packages = 'randomForest') %dopar%

randomForest(infocom06_DC_1 ~User1 + User2 +
ConnStartTime + ConnEndTime + InderxVal + ConnID
+TimeDiff+IndexValOne+ConnIDZero,
data=Infocom06_sample, importance=TRUE, proximity=TRUE,
ntree=ntree)

stopCluster(cl)

Out come

prune(dec_tree, cp=0.006)
n= 227657

node), split, n, loss, yval, (yprob)
      * denotes terminal node

 1) root 227657 101152 No (0.5556825 0.4443175)
```

```
  2) ConnEndTime< 58206 44709  15807 No (0.6464470 0.3535530) *
  3) ConnEndTime>=58206 182948  85345 No (0.5335013 0.4664987)
    6) ConnID>=705.5 73195  29274 No (0.6000546 0.3999454) *
    7) ConnID< 705.5 109753  53682 Yes (0.4891165 0.5108835)
     14) ConnID< 5.5 18618   7312 No (0.6072618 0.3927382)
       28) User2< 3502.5 17607   6301 No (0.6421310 0.3578690) *
       29) User2>=3502.5 1011      0 Yes (0.0000000 1.0000000) *
     15) ConnID>=5.5 91135  42376 Yes (0.4649805 0.5350195) *
```

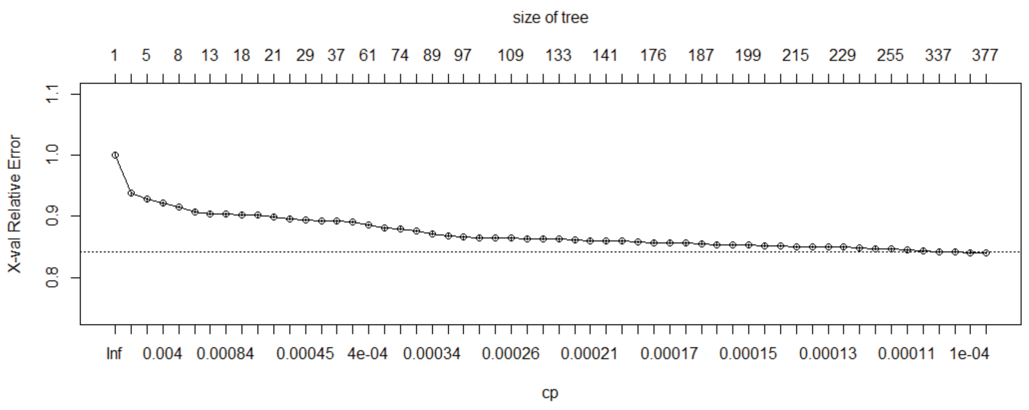


Figure 4.9: Decision tree pruning

4.4 Conclusion

Modelling classification and regression for sybil detection is a very challenging task. Existing research has only made partial progress toward modeling classification for sybil detection and prediction. This research has proposed incremental progress for how sybil activity could be tracked to address this challenging issue. Prediction of sybil behaviour of has been demonstrated by analysing the graph-based classification and regression techniques, using decision trees and described dependencies across different methods. Calculated gain and maxGain helped to trace some sybil users in the datasets. Decision trees were generated in R using our designed algorithm for each node. The results were compared to trees generated by WEKA’s inbuilt C4.5 algorithm to help evaluate and refine our algorithm.

Chapter 5
Conclusion and Future Work

5.1 Conclusion

This thesis presents several techniques to address sybil attacks on mobile social networks using graph, classification and regression analysis.

Chapter 1 introduce the basic concepts of the MSN, social media and data mining concepts including their threats and part of sybil user and its terminology. The data mining classification and regression technique also introduced and explained in this chapter relating to this research.

Chapter 2 provides a literature review of sybil detection in mobile social networks, models and techniques used in past researches. The review summaries the variety of methods and techniques researchers have used to identify sybil attackers in data mining and Big Data models. Variety of models, classification and regression analysis techniques used for decision trees and prediction. Mobile network studies was reviewed for methods of identifying sybil connections and patterns.

Chapter 3 explains content analysis as a general set of techniques useful for analysing and understanding collections of text. Despite its superior accuracy, content analysis is very time-consuming and often demands an impractical level of manual work. Instead, connection analysis can be used to identify the specific connections between two or more nodes in a database and help to discover the nodes, understand how they were established, and describe the behaviour between each connection. Connection analysis draws heavily on the duration and frequency of contact with other nodes. The results may not be as accurate, but the process of identifying sybil users is much faster, and provides a viable and speedy alternative for administrators with limited resources. The Infocom06 dataset of users was used as testbed to identify suspicious user connections. Preliminary results were evaluated using a graph-based system, and showed that sybil users can indeed be distinguished from honest users and evaluated using a graph-based system. The graphs also demonstrate that some sybil users have the highest connectivity to known trustworthy entities. Further experiments, based on content and connection analysis, sought substantial proof for this research outcome using more assumptions and classification and regression techniques. Decision trees were produced for further evaluation.

Chapter 4 proposed a new model based on classification and regression techniques. The dataset was first divided into segments for convenience and to avoid memory issues. The selected attributes for decision making further led to the calculation of an entropy model for each node in the divided dataset. The frequency of connection was then calculated for each node pair. Entropy was calculated based on the three selected column for identifying sybil activity - time difference, index value and connection ID.

The classification model helped to build a decision tree using the C4.5 algorithm, while the calculated entropies contributed to the elimination of nodes considered to be honest. As the steps of entropy calculation moved time difference to connection id of, the processing time also increased. Adjustments were made to the entropy calculation algorithm to increase processing efficiency and honest nodes were skipped during looping. Further, the entropy calculations for each node helped to determine the remaining information gain to be processed for gain calculation. Most of the values in stages two and three were 0, so the gain also fell to 0. Only the gain for level 1 was processed, and it was calculated based on the entropy time difference data. In next phase the maxGain was calculated, which processed a single row for each of 98 users. A random forest was built and the Gini index was calculated. Processing the complete dataset produced 500 trees in the forest. The split resulted in 45% of the data with a time difference, and 55% with no time difference at all. This was a surprising result.

The data was further split into development and validation samples. Both contained a similar target variable distribution. Error rates were plotted across the decision trees. A variable importance plot measured the decrease in the Gini index for the top five variables, again based on time difference, and the accuracy of the dataset was assessed using a confusion matrix which is almost 78% and good and calculated response variable accuracy 59% which still significantly considered based on the dataset (showed in section 4.3.1 and 4.3.2 of this research).

Lastly, the tree was pruned and a graph plotted. An overall comparison, based on Chapters 3 and 4, along with observation of the charts and the behaviour of the classification model for Users 1, 4, 8, 12, 16, 18, 19, 42 and 66 was able to predict the behaviour of sybil users. The predictions were formulated by comparing the tree generated using R with a tree generated by WEKA. Analysis shows that the trees that mostly fall on the right side have negative leaves and a higher value of suspicious entropy compared to other leaves at the same level. This observation provides confidence that the research results are reasonably accurate, and experimentally prove how and why sybil attacks can be modelled for classification.

5.2 Future Work

Based on current predictions, some honest nodes are categorised as sybil attackers. Future research will continue to investigate and refine node identification in mobile social networks.

Random forest processing and the Hadoop system could also be further explored. Generating a random forest using a scoring model via cascading, and its deployment within a Hadoop system are natural next steps. To avoid re-writing the random forest in another programming language, each node must be observed during processing and each incorrect step faithfully logged. Cascading would help define the pipeline processing *pmml* file and enable filtering of the observations through a model.

The approach processed in R software does not currently consider several crucial details, like memory issues and tree production. Coding a new algorithm in Python or Jython may enable the building of an individual model for each user, based on their connection time, for many users simultaneously.

Future studies will also elaborate on the parallelised model building technique: using training data; grouping observations based on users; and generation of a behavioural model for each group.

Incorporating naïve Bayes and k-nearest neighbour techniques would also increase the scope of this research.

User1	User2	freq
1	3	75
1	4	114
1	5	210
1	14	92
1	15	1
~		

Table 4.5 Total count– frequency count for each pair.

User1	User2	TimeDiff	freq
1	3	No	24
1	3	Yes	51
1	4	No	66
1	4	Yes	48
1	5	No	90
1	5	Yes	120
1	14	No	70
1	14	Yes	22
1	15	No	1
~			

Table 4.6 TimeDiff count – frequency count for each pair

User1	User2	TimeDiff	IndexValOne	freq
1	3	No	No	23
1	3	No	Yes	1
1	3	Yes	No	51
1	4	No	No	66
1	4	Yes	No	47
1	4	Yes	Yes	1
1	5	No	No	90
1	5	Yes	No	119
1	5	Yes	Yes	1
1	14	No	No	69
1	14	No	Yes	1
1	14	Yes	No	22
1	15	No	Yes	1
~				

Table 4.7 IndexOneVal count – frequency count for each pair

User1	User2	TimeDiff	IndexValOne	ConnIDZero	freq
1	3	No	No	No	23
1	3	No	Yes	Yes	1
1	3	Yes	No	No	51
1	4	No	No	No	66
1	4	Yes	No	No	47
1	4	Yes	Yes	Yes	1
1	5	No	No	No	90
1	5	Yes	No	No	119
1	5	Yes	Yes	Yes	1
1	14	No	No	No	69
1	14	No	Yes	Yes	1
1	14	Yes	No	No	22
1	15	No	Yes	Yes	1

Table 4.8 ConnIDZero count – frequency count for each pair

U1L1	U2L1	Fq1	EntropyTimeDiff
1	3	75	-0.14769
1	4	114	0.068946
1	5	210	-0.06254
1	14	92	0.193602
1	15	1	NA

Table 4.9 Entropy Calculations – User-TimeDiff datasets for each pair

User1	User2	freq
7	18	2
7	19	2
7	20	7
7	21	2
7	22	1
7	23	4
~		

Table 4.10 Frequency mismatch – with total count for each pair

User1	User2	TimeDiff	freq
7	18	No	2
7	19	No	2
7	20	No	7
7	21	Yes	2
7	22	No	1
7	23	No	3
7	23	Yes	1
~			

Table 4.11 Frequency mismatch – total count with Time Diff for each pair

U1L2	U2L2	TDF2	Fq2	EntropyIndexVal
1	3	No	24	0.132198
1	3	Yes	51	0
1	4	No	66	0
1	4	Yes	48	0.086613
1	5	No	90	0
1	5	Yes	120	0.045585
1	14	No	70	0.067099
1	14	Yes	22	0
1	15	No	1	0
~				

Table 4.12 TimeDiff vs IndexValOne entropy for each pair

U1L3	U2L3	TDF3	IXD3	Fq3	EntropyConnID
1	3	No	No	23	0
1	3	No	Yes	1	0
1	3	Yes	No	51	0
1	4	No	No	66	0
1	4	Yes	No	47	0
1	4	Yes	Yes	1	0
1	5	No	No	90	0
1	5	Yes	No	119	0
1	5	Yes	Yes	1	0
1	14	No	No	69	0
1	14	No	Yes	1	0
1	14	Yes	No	22	0
1	15	No	Yes	1	0
~					

Table 4.13 IndexValOne vs ConnIDZero entropy for each pair

U1L3	U2L3	TDF3	IXD3	Fq3	EntropyConnID
55	34	No	No	11	0.189491
55	37	No	No	10	0.19539
55	40	No	No	19	0.149678
55	43	Yes	No	7	0.210429
55	48	No	No	8	0.206436
55	60	No	No	11	0.189491
55	60	Yes	No	11	0.189491
55	66	No	No	5	0.206843
55	76	No	No	11	0.189491

Table 4.14 IndexValOne vs ConnIDZero few entropy for User 55

U1L1	U2L1	Fq1	EntropyUser	Gain
1	3	75	-0.14769	-0.10538
1	4	114	0.068946	0.032477
1	5	210	-0.06254	-0.08859
1	14	92	0.193602	0.244655
1	15	1	NA	0
~				

Table 4.15 Information gain calculations

User1	User2	Frequency	Entropy	Gain	MaxGain
1	41	7	0.210429	0	0.391827
2	457	7	0.210429	0	0.391827
3	23	7	0.210429	0	0.391827
4	146	7	0.210429	0	0.391827
5	157	7	0.210429	0	0.391827
~					
94	98	7	0.210429	0	0.391827
95	16	7	0.210429	0	0.391827
96	13	7	0.210429	0	0.391827
97	59	6	0.211632	0	0.384001
98	42	7	0.210429	0	0.391827

Table 4.16 maxGain calculations

User1	User2	Frequency	Entropy	Gain	MaxGain
1	41	7	0.210429	0	0.391827
4	146	7	0.210429	0	0.391827
8	44	9	0.201169	0	0.384668
12	386	6	0.211632	0	0.384001
16	211	7	0.210429	0	0.391827
18	136	7	0.210429	0	0.391827
19	290	7	0.210429	0	0.391827
42	39	7	0.210429	0	0.391827
66	63	7	0.210429	0	0.391827

Fig. 4.17: Sybil users information gain

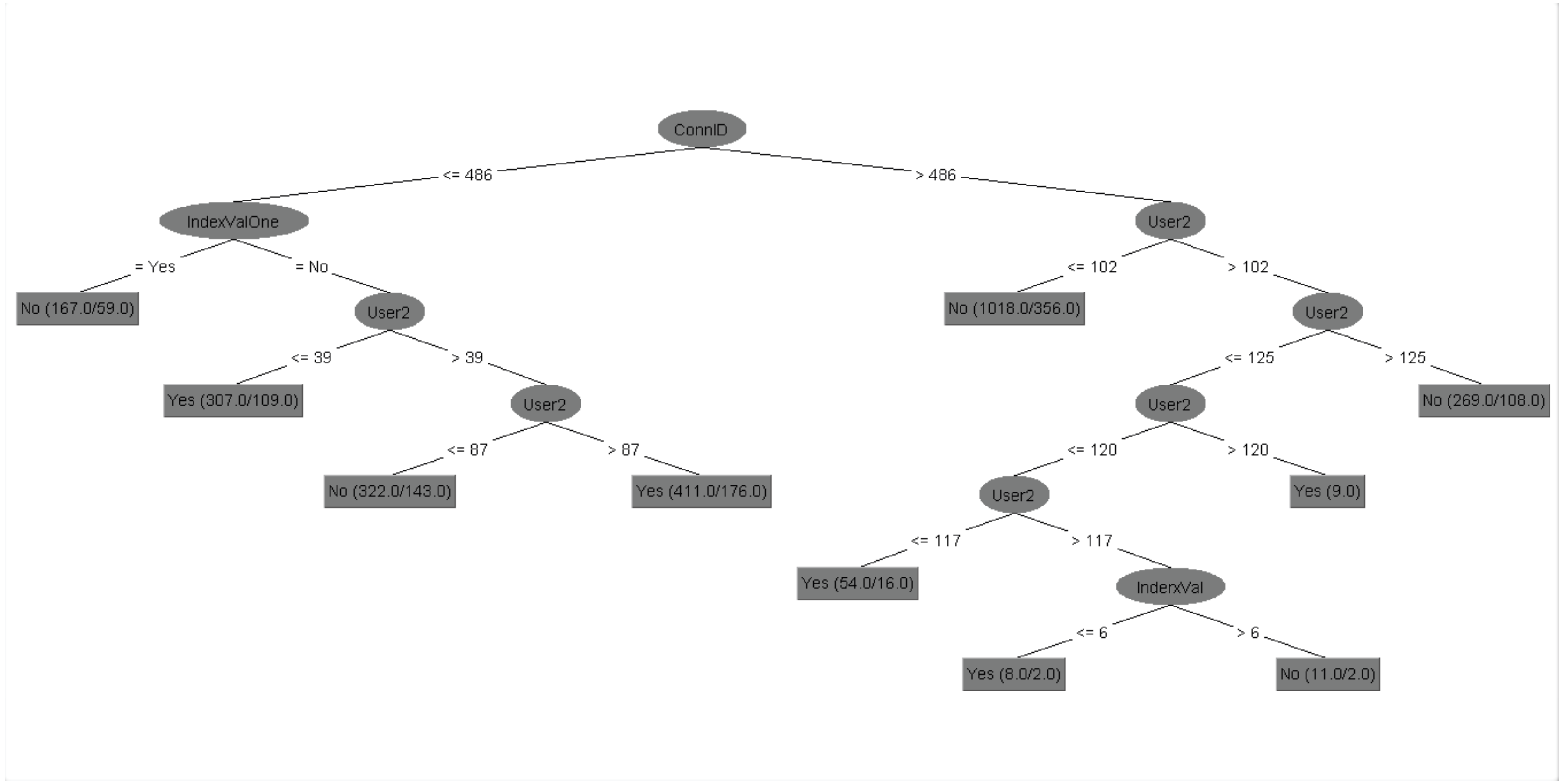


Fig. 4.10: User 1 decision tree

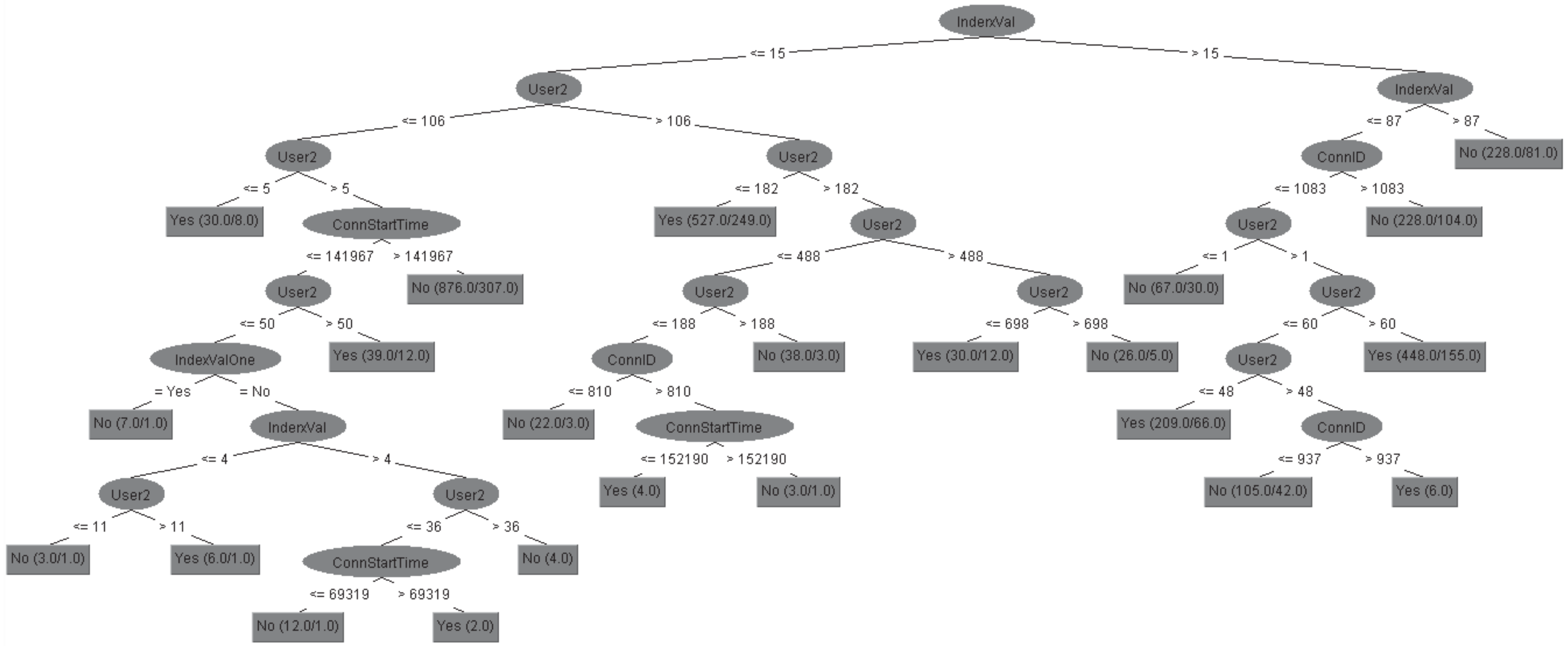


Fig. 4.11: User 4 decision tree

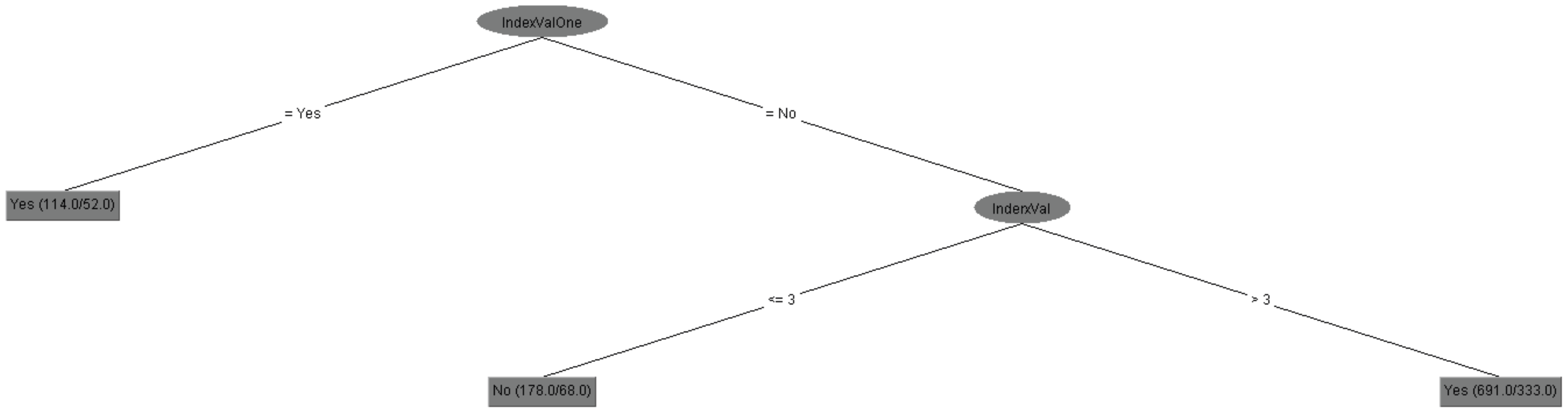


Fig. 4.12: User 8 decision tree

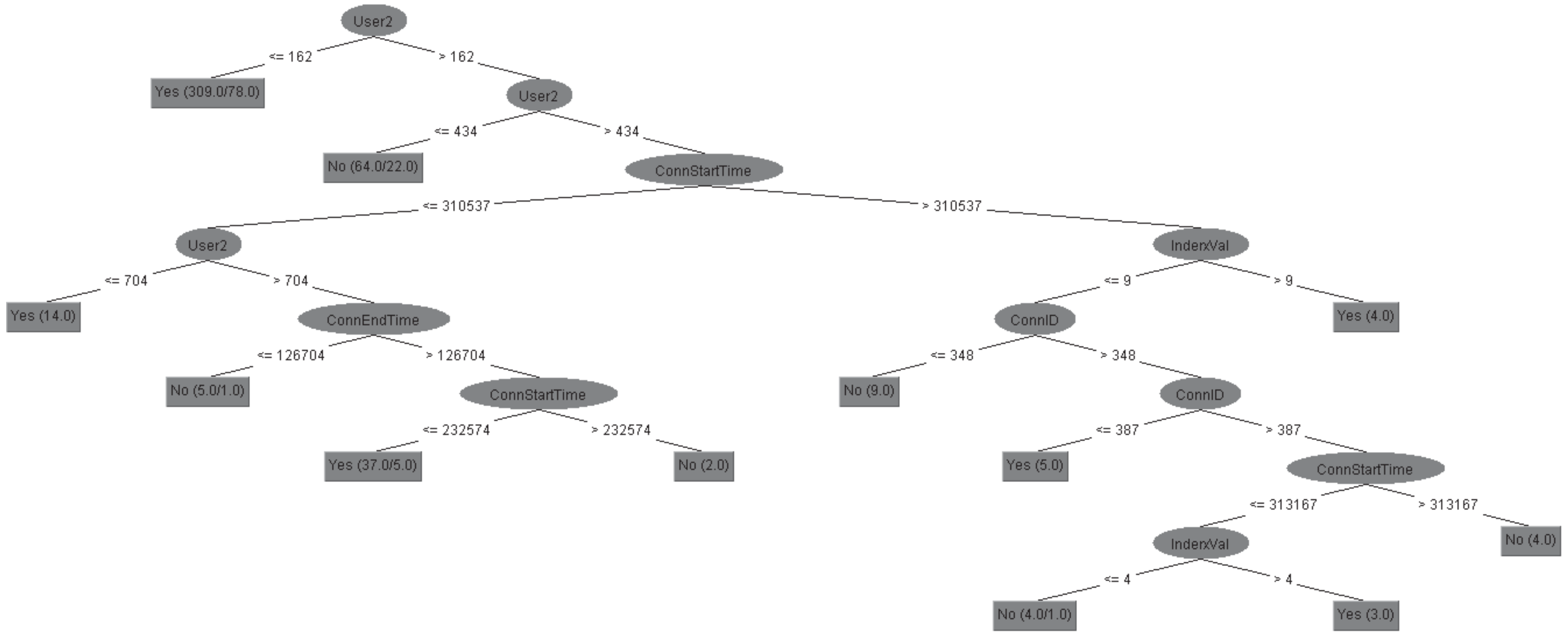


Fig. 4.13: User 12 decision tree

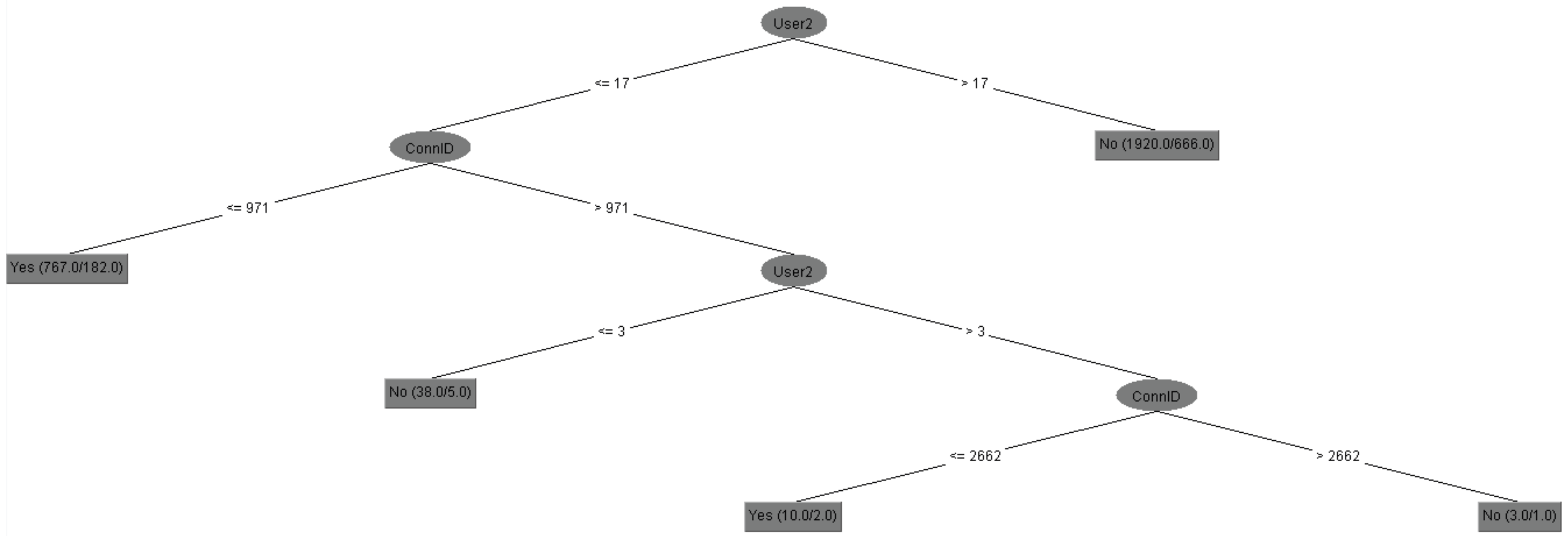


Fig. 4.14: User 16 decision tree

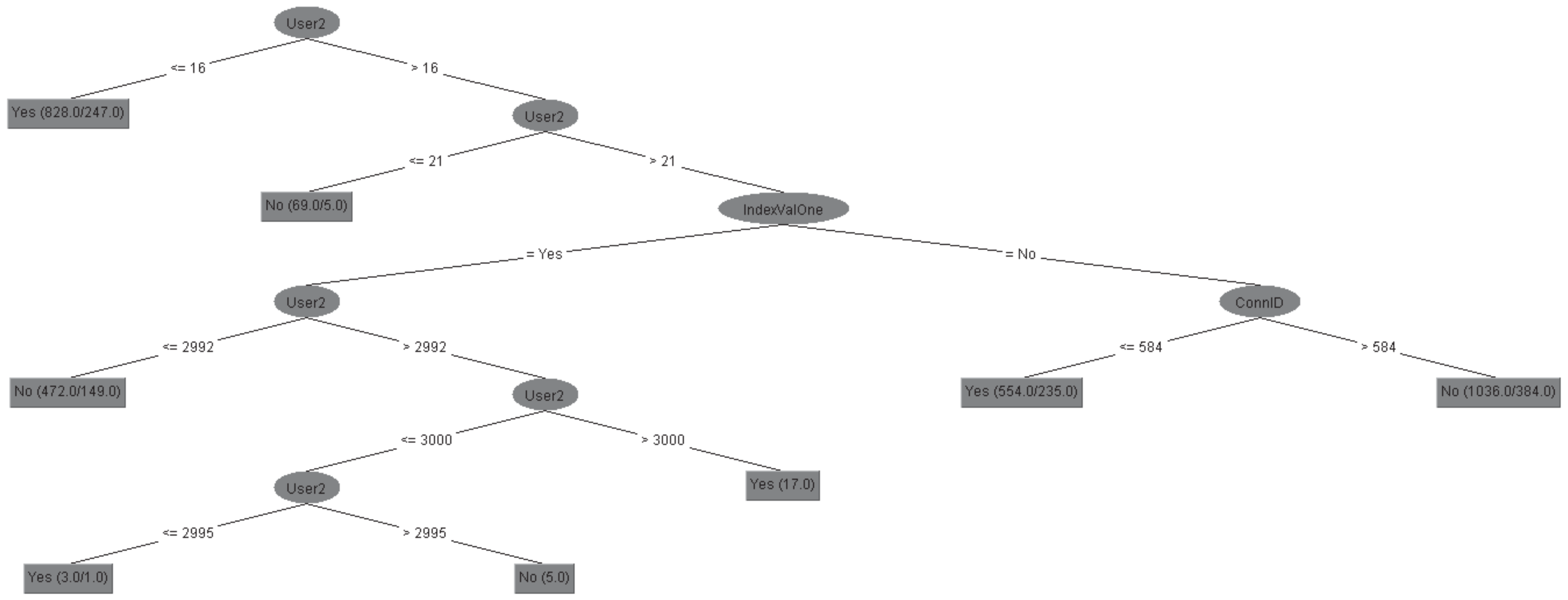


Fig. 4.15: User 18 decision tree

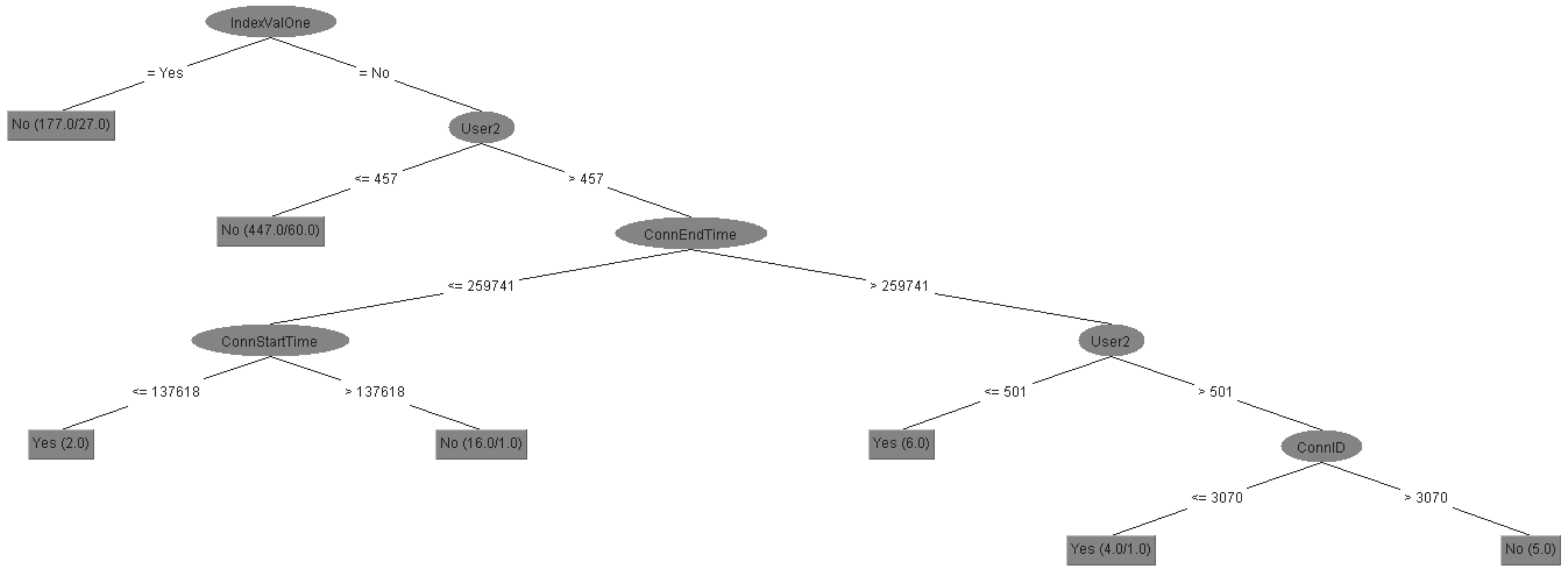


Fig. 4.16: User 19 decision tree

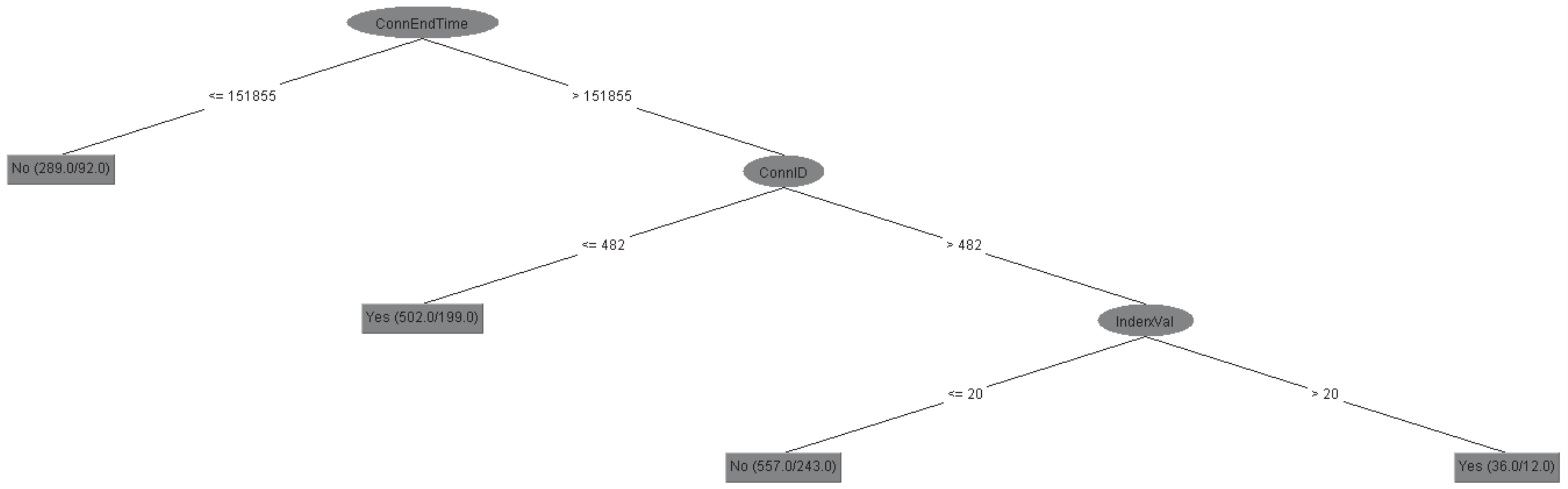


Fig. 4.17: User 42 decision tree

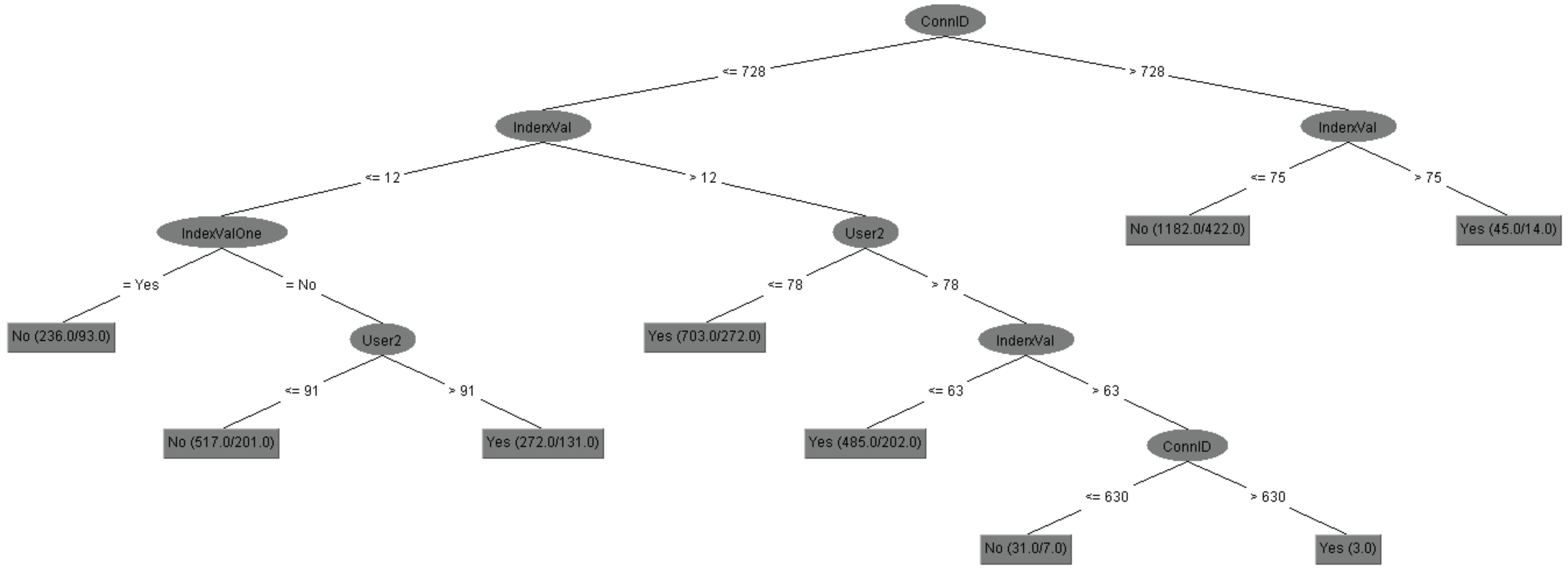


Fig. 4.18: User 66 decision tree

Appendix

List of Symbols

H(S)	Entropy
S	Training set / subset training example
p(+)	Positive examples in S (total number of purity)
p(-)	Negative examples in S (total number of impurity)
V	Possible or particular values of A
Sv	Subset X_A= V which is all the fixed time or all the maximum connections
A	Attribute
$P(p_1, p_2 \dots, p_N)$	Probability distribution
S_N	Set of all Complete Probability distribution
Hn	Honest Entity
Sn	Sybil Entity
∂	Condition to check
TD	Output of Total Time Difference
ST	Starting time
ED	Ending time
S1, S2...	Different Training set
x_j	Represent attribute values or features of the sample
C	Set of training instances
P	Class of Training Instance
fi	Number of branches
Spl	Each split

Sv	Subset where X_A= V
$S_{Suspicious}$	Suspicious Training Set
S_{Honest}	Honest Training Set
N1, N2, N3, N4	Calculated length of dataset
Data1, Data2, Data3	Data frames to store new data

Bibliography

Abaya, S. A., Gerardo, B. D., ' An education data mining tool for marketing based on C4.5 classification technique', e-*Learning and e-Technologies in Education (ICEEE), Second International Conference,* pp. 289-293 (2013)

Alvisi, L., Clement, A., Epasto, A., Lattanzi, S., Panconesi, A., 'SoK: The Evolution of Sybil Defense via Social Networks : Security and Privacy (SP)' , 2013: In: *IEEE Symposium on security and privacy*, pp. 382 -396 (2013)

Behera, G., 'Privacy preserving C4.5 using Gini index', *Emerging Trends and Applications in Computer Science (NCETACS), 2nd National Conference*, pp. 1-4 (2011)

B S, J. and D. Janakiram, 'SyMon: A practical approach to defend large structured P2P systems against Sybil Attack', *Peer-To-Peer Networking and Applications*, vol- 4(3),: pp. 289-308 (2011)

Cortimiglia, M. N., Renga, F., Ghezzi, A, 'Mobile Social Networking: A Case Study in an Australian Mobile Network Operator, Mobile Business (ICMB)', 2011 *Tenth International Conference on Mobile Business*, pp. 84-92 (2011)

Haifeng, Yu, Gibbons, P. B., Kaminsky, M., Feng, X., 'Sybil limit: A Near-Optimal Social Network Defense Against Sybil Attacks', *Networking, IEEE/ACM Transactions*, vol- 18(3), pp. 885-898 (2010)

Haifeng, Yu, Kaminsky, M., Gibbons, P. B., Flaxman, A. D., ‘Sybil guard: Defending Against Sybil Attacks via Social Networks’, *Networking, IEEE/ACM*, vol -16(3), pp.576-589 (2008)

Hao, X., X. Weidong, Tang, D., Tang, J., Wang, Z., ‘Core-based community evolution in mobile social networks’, *Big Data , IEEE International Conference* (2013)

Huiqi, Z. and Dantu, R., ‘Predicting social ties in mobile phone networks’, *Intelligence and Security Informatics (ISI)*, 2010 IEEE International Conference, pp. 25-30 (2010)

Huiqi, Z., Dantu, R., Cangussu, J. W., ‘Socioscope: Human Relationship and Behavior Analysis in Social Networks’ , *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions*, vol- 41(6), pp. 1122-1143 (2011)

Jing, J., Zifei, S., Wenpeng, S., Xiao, W., Yafei, D., ‘Detecting and Validating Sybil Groups in the Wild’, *Distributed Computing Systems Workshops (ICDCSW)*, 32nd International Conference, pp. 127132 (2012)

Koleshwar, A., Thakare, V., Sherekar, S. ‘Study of Mobile Cloud Computing Security against Cyber Attacks’, *International Journal of Advanced Research in Computer Science*, vol- 5(4), (2014)

Mei, A. and J. Stefa, ‘ Give2Get: Forwarding in Social Mobile Wireless Networks of Selfish Individuals’*, Dependable and Secure Computing, IEEE Transactions,* vol - 9(4): pp. 569-582 (2012)

Mohaien, A., D. F. Kune, ‘Secure Encounter-Based Mobile Social Networks: Requirements, Designs, and Tradeoffs: Dependable and Secure Computing’ , *IEEE Transactions*, vol-10(6): pp. 380-393 (2013)

Liang, X., Li, X., Zhang, K., Lu, R., Lin, X., Shen, X., 'Fully Anonymous Profile Matching in Mobile Social Networks', *IEEE Journal on Selected Areas in Communications*, vol- 31, (9): pp. 641-655 (2013)

Liu, P., Wang, X., CHe, X., Chen, Z., Gu, Y., 'Defense against sybil attacks in directed social networks', *Digital Signal Processing (DSP), 19th International Conference* (2014)

Quercia, D. and Hailes S., 'Sybil Attacks Against Mobile Users: Friends and Foes to the Rescue', *INFOCOM, Proceedings IEEE* (2010)

Shrivastava, N., Majumder, A., Rastogi, R., 'Mining (Social) Network Graphs to Detect Random Link Attacks', *Data Engineering, ICDE, IEEE 24th International Conference*, pp. 486-495c (2008)

Wang, G., Konolige, T., Wilson, C., Wang, X., 'You Are How You Click: Clickstream Analysis for Sybil Detection', *Proceedings of the 22nd USENIX conference on Security (USENIX Security)*, pp. 241-256 (2013)

Wei, C., Jie, W., Tan, C. C., Feng, L., 'Sybil defenses in mobile social networks', *Global Communications Conference (GLOBECOM)*, IEEE, pp. 641-646 (2013)

Wei, W., Fengyuan, X., Tan, C. C., Qun, L., 'SybilDefender: Defend against sybil attacks in large social networks', *INFOCOM, 2013 Proceedings IEEE*, pp. 1951 -1959(2013)

Xiaohui, L., L. Xiaodong, Shen, X., 'Enabling Trustworthy Service Evaluation in Service-Oriented Mobile Social Networks: Parallel and Distributed Systems', *IEEE Transactions,* vol -25(2): pp. 310-320 (2014)

Xu, G., Zhang, Y., Zhou, X., 'Discovering Task-Oriented Usage Pattern for Web Recommendation', *Sevents Australiasian Database Conference (ADC) (CRPIT)*, 2006, on vol – 49 (2006)

Yan, S., Lihua, Y., Wenmao, L., 'Defending sybil attacks in mobile social networks', *Computer Communications Workshops (INFOCOM WKSHPS)*, *IEEE Conference*, 163-164 (2014)

Yang, Z., Wilson, C., Wang, X., Gao, T., Zhao, B. Y., Dai, Y., 'Uncovering social network sybils in the wild', *Proceedings of the ACM SIGCOMM conference on Internet measurement conference*, pp. 259-268 (2011)

Yuhang, Z., Zhaoxiang, Z., Yunhong, W., Jianyun, L., 'Robust mobile spamming detection via graph patterns' , *Pattern Recognition (ICPR)*, *21st International Conference*, pp. 127 -132 (2012)

Zhang, B., Y. Wang, Vasilakos, A., V., Ma, J., 'Mobile social networking: reconnect virtual community with physical space', *Telecommunication Systems,* vol -54(1): pp. 91-110 (2013)

Zhang, K., Liang, X. Lu, R., Shen, X., 'SAFE: A social based updatable filtering protocol with privacy-preserving in mobile social networks', *Communications (ICC), IEEE International Conference*, pp. 6045 -6049 (2013)

Zhang, K., Liang, X. Lu, R., Shen, X., 'Sybil Attacks and Their Defenses in the Internet of Things', *Internet of Things Journal, IEEE,* vol.- 1(5), pp. 372-383 (2014)

Zhang, K. Liang, X. Lu, R. Yang, K., Shen, X., 'Exploiting mobile social behaviors for Sybil detection'*, IEEE Conference on Computer Communications (INFOCOM), Kowloon*, pp. 271-279 (2015)

Zhang, X., Zheng, H. Li, X., Du, S., Zhu, H., 'You are Where You Have Been: Sybil Detection via Geo-location Analysis in ONSs', *IEEE Global Communications Conference, Austin, TX*, pp. 698-703 (2014)