Learning Decision Trees from Synthetic Data Models for Human Security Behaviour

Peter Carmichael[1] and Charles Morisset[1]
p.j.carmichael@ncl.ac.uk charles.morisset@ncl.ac.uk

Newcastle University, Newcastle upon Tyne, United Kingdom

Abstract. In general, in order to predict the impact of human behaviour on the security of an organisation, one can either build a classifier from actual traces observed within the organisation, or build a formal model, integrating known existing behavioural elements. Whereas the former approach can be costly and time-consuming, and it can be complicated to select the best classifier, it can be equally complicated to select the right parameters for a concrete setting in the latter approach. In this paper, we propose a methodical assessment of decision trees to predict the impact of human behaviour on the security of an organisation, by learning them from different sets of traces generated by a formal probabilistic model we designed. We believe this approach can help a security practitioner understand which features to consider before observing real traces from an organisation, and understand the relationship between the complexity of the behaviour model and the accuracy of the decision tree. In particular, we highlight the impact of the norm and messenger effects, which are well-known influencers, and therefore the crucial importance to capture observations made by the agents. We demonstrate this approach with a case study around tailgating. A key result from this work shows that probabilistic behaviour and influences reduce the effectiveness of decision trees. This impact is in regards to error rate, precision and recall measurements.

Keywords: Synthetic data · Human behaviour · Decision trees · Security

1 Introduction

Employees of organisations are known to regularly circumvent or bypass security procedures, leading to a relaxed security culture [1]. In order to identify the security culture of an organisation, a security practitioner could collect data from different sources and build a classifier model to predict the security preference of employees. For example, sources such as CCTV, interviews and physical logs (smart card data) can be used to classify employees preferences. There are three main challenges with this - 1) It is costly both in time and financially, as demonstrated by Caufield and Parkin [4]. 2) It is error prone as we rely on humans to interpret human behaviour. 3) Given the dataset, it is difficult to identify which features are relevant to build a classifier model.

To address the three challenges, we propose an assessment of classifier models known as decision trees to predict the impact behavioural elements have on the security culture of an organisation. Firstly, we generate synthetic data from parameterised models with behavioural elements. Secondly, we interpret the data to assess features based on the dataset. Finally, using traditional data mining techniques, we use cross validation to train and test each model independently. One of our results shows that different parameters for human behaviour impact decision trees differently. Probabilisitc behaviour clearly impacts the error rate, precision and recall of decision trees the most.

Our approach is inspired by online marketing techniques, where peoples behaviour is logged and suggestions are made based on purchases of others who have similar behaviour trends [21]. In the context of security, building a classifier model, an employee performing a security violation should be more substantial to the models rules than the same employee moving between locations. Identifying relevant features, indicates that human behaviour is required, in some form at least. Influences, such as *Social Proof* from Cialdini, or the *Messenger* effect from MINDSPACE can shift the culture of an organisation [7,8]. An employee in the right location at the right time may be influenced by the behaviours of others. If they observe the same action multiple times, or have an influential relationship with the instigator of the behaviour, then they may begin to change their behaviour and act accordingly.

We analyse a case study surrounding tailgating where we simulate a number of parameterised models to identify the accuracy of behavioural elements on decision trees. We believe that a security practitioner can understand the relevant features and can begin to understand the relationship between the complexity of human behaviour and the accuracy of decision trees. As a security practitioner, acquiring knowledge where vulnerabilities are present, allows for insights towards defence strategies and interventions. For example, investing in turnstiles to reduce tailgating, or limiting the capabilities of employees who are flagged as a vulnerability.

The main contributions of this work are 1) Parameterised models to simulate and generate synthetic data with known behavioural elements. 2) The methodology for the assessment of precision and recall for decision trees constructed from a synthetic dataset.

The paper is split into the following Sections. In Section 2 we discuss the problem, provide an intuition for how we are approaching it and build on existing literature. In Section 3 we introduce our Multi Agent System (MAS) to generate synthetic data. In Section 4 we discuss the Case Study and the parameters used. In sections 5 and 6 we discuss our assessment methodology and analyse the case study. Section 7 is the conclusion and future work.

2 Problem Formulation

A security practitioner is able to observe employees behaviour in an organisation and accumulate information about security incidents. This data can form a trace,

where an entry in a trace describes who did what and when. It is similar to an intrusion detection system, where the logs of what happened are the entries, a collection of logs/entries forms a trace. Consider a simple probabilistic model

Agent	**Violations**	**Preventions**	**Security Preference**
Alice	4	1	*Usable*
Bob	2	3	*Secure*
Charlie	1	0	*Usable*
Dan	2	5	*Secure*

Table 1: Tallied up data for user actions

with agents, where agents can be *secure* or *usable*. Agents who are *secure* have a probability of 0.2 to perform a security violation, whereas *usable* agents have a probability of 0.8 to perform a violation. When a violation occurs, it involves another agent, where the probabilities for them to permit the violation are 0.2 and 0.8 for *secure* and *usable* respectively. A security practitioner observes agents in an organisation performing actions and accumulates information about each agent. The security practitioner is only able to collect three features for an agent; The number of violations, preventions and each agents security preference.

Using the data from Table 1, which is the accumulation of agent actions we wish to establish some form of learning process, one example is a decision tree. By doing this, we want to learn whether or not it is feasible to use classifier models, such as decision trees to predict and identify when agents behave insecurely.

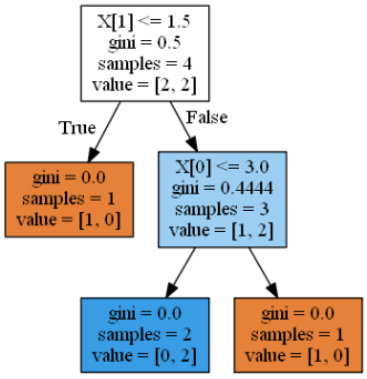


Fig. 1: Simple Decision Tree - X[1] = Preventions, X[0] = Violations

Figure 1 is an example tree that we can learn from the data in Table 1. Where the tree would take some test data, without the security preference and attempt to determine it using the conditional rules of each node. Obviously, this is a small sample, but an accurate tree has been synthesised with 100% accuracy for the training data used to create it. An issue would arise with an employee who has no observed preventions or violations, the decision tree will always return

them as *usable*. Additionally, a combination of behaviour features that makes the rules worthless, would ensure that predictions are innacurate,

For a security practitioner, establishing which features to consider could provide meaningful results for observing the security culture of employees in an organisation. Unfortunately, the problem is of greater complexity than what we identify here, as human behaviour is complex in itself and a leads us to the question, can we learn from complex behaviour?

2.1 Security Culture Uncertainty

Within an organisation, there is a security culture for how individuals and groups of people respond to security incidents. Depending on the type of security incident and the people involved, it could become a security violation or it could be prevented. We hope that individuals trained to perform tasks are security aware, but we regularly find that they circumvent organisational security policies [3].

For a security officer, explaining and classifying the security culture of an organisation is a complex task. Consider working with a company for a short period of time in order to identify the security preference of employees. We could ask them, where responses from interviews have lead to popular theories such as the compliance budget [1]. Of course, respondents could lie, answer honestly but not behave consistently, or even fail to acknowledge that their behaviour is insecure.

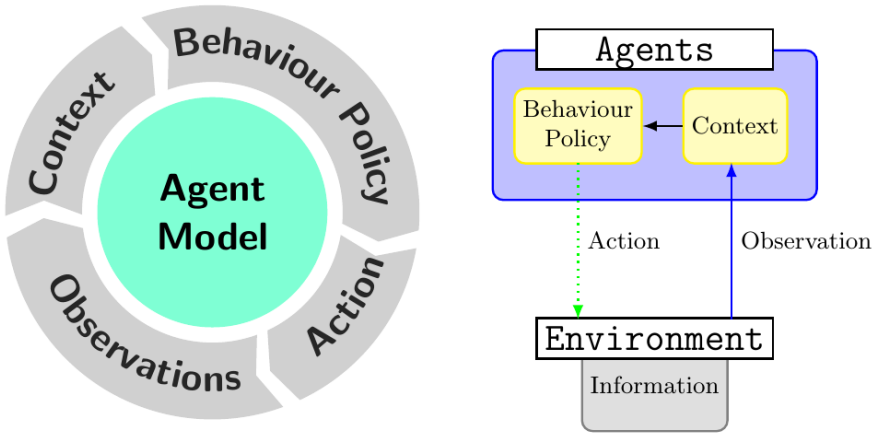
Even if survey respondents answer honestly, it does not mean that this holds for the future. A secure employee interacting with a usable (non-secure) employee may be influenced towards usable behaviours, creating an insecure culture. Of course, this is bi-directional where secure behaviour can inform more secure behaviour.

From a security officers perspective, they only have so many tools to establish the security culture. For example, they could interview employees, then manually observe them via CCTV recordings to establish if their security preference matches their behaviour [4]. This is of course, costly and time consuming, where we would need to manually record the exact behaviour of each employee. This would not allow us to classify the behaviour of agents that have been interviewed and not observed.

To add further complexity to the uncertainty of a security culture, some one who is secure may make a judgement of error causing a security incident. For example, Zhu *et. al.* showed they could get more information from people simply by providing them with information up front, exploiting a concept known as reciprocity [23]. They were able to influence people to sacrifice more information than they usually would part with.

The security culture of a company can be changed, for example, via training employees [17]. This behaviour change is one that impacts how people respond to security incidents, for example a recently trained employee may have more awareness for *spear phishing* emails, and is less likely to click suspicious links.

3 Multi Agent System



(a) Ordering of agent interactions

(b) Agent Interaction with Environment

Fig. 2: MAS - Agent Behaviour

To identify the security culture of an organisation, we must first acquire data for user behaviours with security incidents. To the best of our knowledge this data does not exist, therefore, we must either, take a different approach to solve our research question, or generate our own synthetic data. We use insights from computer security, human decision making, behaviour change theory and economics and define a MAS as presented in Sections 3.1-3.4 [13,12,8,16].

3.1 Agents - Actions

The literature for modelling agents is captured in a variety of different manners. From economics, we see humans referred to as homo economicus, although the homo economicus has always been comparable to the smartest economist [18]. In simulation, agents with internal beliefs are captured, who act on those beliefs. The BDI (belief-desire-intention) architecture is one example capturing agents mental state to drive actions [14].

We illustrate our agent model in Figure 2 (a) showing the interactions of the agent, whilst providing a sequence for this. Figure 2 (b) expresses the agents interactions with the environment. From this model we will discuss Actions, Observations, Context and Behaviour Policies.

Definition 1 *An agent is a relation and is defined as* $A \subseteq A_{id} \times L \times \Theta \times C$ *where* A_{id} *is the set of agent ids,* L *is the set of locations,* Θ *is a set of observations and* C *is the context. Such that* $(a, l, \Theta_a, c) \in A$ *would indicate that the agent* a *is in the location* l *with the set of observations* Θ_a *and the context* c*.*

We introduce the notion of location to capture a partially observable environment for agents. Agents perform actions which change the environment. For example, the location of agents in an environment can change as the result of an action. In essence, an action can change the state of the environment.

Definition 2 *We define the set of actions as Act and any act* $\in$ *Act is represented in the form of* $\phi \xrightarrow{act} \phi^{'}$. *Where* ϕ *is a pre-condition and* $\phi^{'}$ *is the post condition as a result of act.*

Running Example: We now provide a running example which we use to demonstrate the MAS. Consider four agents *Alice, Bob, Charlie, Dan* where the following holds:

- $Alice = (1, l_1, \{\emptyset\}, 0)$, $Bob = (2, l_1, \{\emptyset\}, 1)$
- $Charlie = (3, l_2, \{\emptyset\}, 0)$, $Dan = (4, l_2, \{\emptyset\}, 1)$

The agents *Alice* and *Bob* are in a different location to *Charlie* and *Dan*. An action rule R1 is $(a, l_1, \Theta_a, c) \xrightarrow{move} (a, l_2, \Theta_a, c)$ which reads that an agent in location l_1 can move to location l_2 and all other elements remain the same.

3.2 Agents - Observations

Actions are observable, where agents can accumulate observations. Typically, in a MAS, agents are capable of observing the environment [22].

Definition 3 *Observations are a relation* $\Theta \subseteq A_{name} \times Act$ *such that if* $\theta_1 \in \Theta$ *and* $\theta_1 = (a_1, act_1)$ *would indicate an observation of the agent* a_1 *performing* $act_1 \in Act$. *We also define* Θ_{obs} *as the current observation, where the current observation is updated based on action rules.*

To accommodate for the current observation we re-write rule R1, as shown below. This ensures that a current observation becomes unavailable for observation once a new action occurs, that itself, is not an observation. An observation rule, like an action rule follows a format. An abstract notion of an observation would be $(a, l, \Theta, c), \Theta_{obs} \xrightarrow{\theta} (a, l, \Theta \cup \theta, c), \Theta_{obs}$ if $\theta \in \Theta_{obs}$. Where it reads the agent a in location l with the observations Θ observes θ and becomes the same agent a in the same location l with the observations $\Theta \cup \theta$.

$$R1 : (a, l_1, \Theta_a, c), \Theta_{obs} \xrightarrow{move} (a, l_2, \Theta_a, c), \emptyset$$
$$R2 : (a, l_2, \Theta_a, c), \Theta_{obs} \xrightarrow{move} (a, l_1, \Theta_a, c), \emptyset$$
$$R3 : (a, l_1, \Theta, c), \Theta_{obs} \xrightarrow{act_1} (a, l_1, \Theta \cup (a, act_1), c), (a, act_1)$$
$$R4 : (a, l, \Theta, c), \Theta_{obs} \xrightarrow{\theta} (a, l, \Theta \cup \theta, c), \Theta_{obs} \text{ if } \Theta_{obs} \neq \emptyset$$

Table 2: Action and Observation Rules:

Consider agent *Alice* executing R3, such that $\Theta_{obs} = (1, act_1)$, by R4, only *Bob* can observe this. If agents *Charlie* or *Dan* executes R2 and enters location

l_1 they cannot observe $(1, act_1)$ as they arrived after act_1 occurred. The only two agents who can observe *Alice* performing act_1 is *Alice* and *Bob* given their current locations.

Agents perceive different observations differently dependent upon their current context, where the context is the internal state of an agent [13,5]. Consider two agents, one whose context is *usable*, the other is *secure* towards security violations. The *usable* agent does not acknowledge a security violation. Whereas, the *secure* agent's context informs that a security violation has occurred, as such, they respond differently. Let us define that 0 and 1 capture usable and secure respectively.

$$R5 : (a_1, l_2, \Theta_1, c_1)(a_2, l_2, \Theta_1, c_2), \Theta_{obs} \xrightarrow{violation}$$
$$(a, l_2, \Theta \cup \theta, c), (a_2, l_2, \Theta_1, c_2)\{(a_1, violation), (a_2, permit)\} \text{ if } c_1 = 0 \text{ and } c_2 = 0.$$
$$R6 : (a_1, l_2, \Theta_1, c_1)(a_2, l_2, \Theta_1, c_2), \Theta_{obs} \xrightarrow{prevented}$$
$$(a, l_2, \Theta \cup \theta, c), (a_2, l_2, \Theta_1, c_2), \{(a_1, prevented), (a_2, denied)\} \text{ if } c_1 = 0 \text{ and } c_2 = 1$$

Table 3: Action Rules - Security Violations

The rule R5 states that a violation can occur if two agents in l_2 are present and both are *usable*. Whereas, R6 is a prevented rule, where the usable agent attempts to violate and is denied by the secure agent. Let us consider our four agents where *Alice* and *Charlie* are usable and *Bob* and *Dan* are secure. Given their initial state, R5 cannot be executed, where as R6 can be executed. In order for a violation to occur, *Alice* must execute R2 then $R5$ with *Charlie*.

3.3 Agents - Behaviour Change

Without any policies to govern behaviour, agents actions remain consistent, in our case *Alice* and *Charlie* will always violate and *Bob* and *Dan* will always prevent a violation. A static context ensures agents will always perform the same action in the same situation. From a security perspective, this type of model fulfills the requirement to model agents interacting. Where we could answer a question such as *What is the chance of a security breach?*. It doesn't allow us to answer a question such as *What is the change in the security culture of an organisation?*. The second question, requires agents to have the opportunity for dynamic behaviour.

For example, *What change in the security culture significantly increases the likelihood of a security breach?*. Answering a question such as this, leads us to consider concepts such as a tipping point with positive feedback. By this, we mean a small change in the security culture increasing the magnitude of the change, which recurrently impacts the system. A classic example of a positive feedback loop is a bank run, where worried customers withdraw all savings from a bank, influencing others to withdraw their own funds [11]. In our application, it would be usable agents influencing secure agents to become usable, creating

a growth in the security culture where more agents then become usable, and so on.

From social psychology, role theory suggests that actions carried out by people follow convention of their role and the expectations that come with it [2]. For example, a security violation of tailgating at a company would expect that someone with the role of guard would stop this from occurring. Attribution theory claims that actions carried out by humans are informed by the observations they make of others [10].

Cumulative Behaviour Change: This has been explored by Caufield *et al.* where they use the notion of a private signal (context) and public signal (observation) to inform decisions [5]. We use it as a basis to model influences as cumulative observations, where a threshold of observations for a particular action must be reached in order to trigger an influence policy. Where a behaviour policy is based on the context and observations of an agent [13].

A behaviour policy in regards to rules would follow the format:

$$(a, l, \{a_1, violation\}, 1), \Theta_{obs} \xrightarrow{policy} (a, l, \emptyset, 0), \Theta_{obs} \text{ if } a \neq a_1$$

which reads, the agent with id a in location l and is secure changes to usable if they have observed a violation by a different agent a_1.

$$R7 : (a_1, l, \{(a_2, violation), (a_3, violation)\}, 1), \Theta_{obs} \xrightarrow{usable} (a_1, l, \emptyset, 0), \Theta_{obs} \text{ if } a_1 \neq a_2 \text{ and } a_1 \neq a_3$$

$$R8 : (a_1, l, \{(a_2, prevention), (a_3, prevention)\}, 0), \Theta_{obs} \xrightarrow{secure} (a_1, l, \emptyset, 1), \Theta_{obs} \text{ if } a_1 \neq a_2 \text{ and } a_1 \neq a_3$$

Table 4: Behaviour Change Rules:

Given the behaviour change rules, where agents are influenced by a set of observations, the security culture of the four agents can shift. Given the initial state consider the sequence of rules:

SQ1: $Alice : R2 \rightarrow R5$, $Dan : R4$, $Charlie : R5$, $Dan : R4 \rightarrow R7 \rightarrow R5$
SQ2: $Bob : R2 \rightarrow R6$, $Charlie : R4$, $Dan : R6$, $Charlie : R4 \rightarrow R8 \rightarrow R6$

Sequence SQ1 would shift the security culture to {0,1,0,0} for *Alice*, *Bob*, *Charlie* and *Dan* respectively. Sequence SQ2 would shift it to {0,1,1,1}. Depending on the order of actions, the shift in the security culture can be impacted.

We use these four concepts of actions, observations, context and behaviour policy to capture our MAS. This is our abstract model that we use a foundation for the rest of the paper.

4 Case Study

In this section, we introduce and illustrate a tailgating case study where agents move between locations and make decisions based on the actions they observe

and any internal behaviour change policy that they are bound by. Figure 3 shows the possible actions an agent can make when they enter the back of the reception.

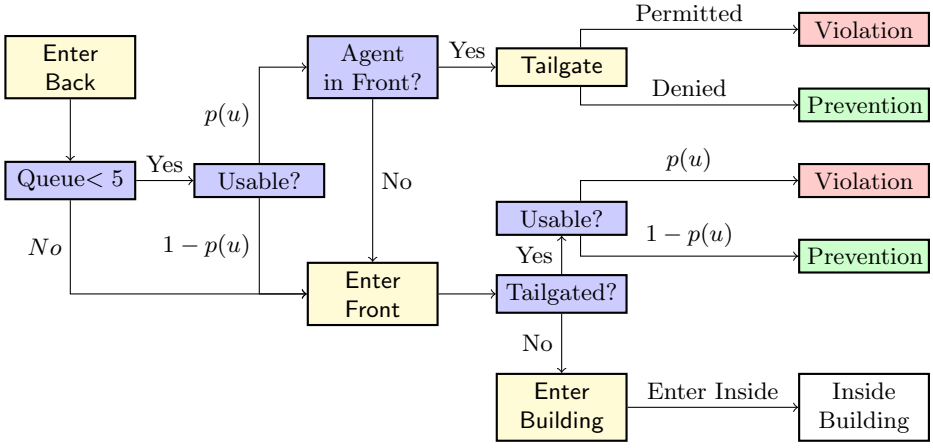


Fig. 3: Agent actions as they enter the reception.

Scenario 1 *Agents arrive at the back of the workplace reception and there are two possibilities. Firstly, if nobody is at the front of the reception, the agent must progress to the front of the reception. Secondly, and dependent upon the agents security preference (usable or secure), if less than five people are at the front of reception and about to enter, and the newly arrived agent is usable, they will attempt to tailgate. A perfectly secure agent will never attempt to tailgate. If an agent is being tailgated, they can either permit or deny the action, where a permit would allow both agents into the main building, a deny would force the tailgater to the front of the reception. A perfectly usable agent will always permit tailgating, a secure agent will deny tailgating. The scenario runs for a working week of five days.*

4.1 Parameters

For the case study, we want to reflect behaviours and attributes that we know exist. We understand that, whilst we will not have a model that truly reflects human behaviour, we at least can parameterise concepts that we know exist from the literature. Our parameters are as follows:

– p_1: Expected Arrival Rate - Agents arrive stochastically to the workplace reception, the arrival rate follows a normal distribution, where agents can arrive at any point within some bounds. For example, if a start time for work is 9AM, we might expect some agents to turn up early, just before, just after, late or preciously on start time.

- p_2: Probabilistic Decision - A lot of assumptions have been made towards individuals as being *homo economicus*, where we make decisions based on personal gain or internal heuristics for guiding behaviour which look to maximise some reward [9]. Additionally, each day, experience is slightly different and for an agent, this could be the difference between a *secure* agent acting *usable* and vice versa, which is what we capture with our probabilistic decision, the ability for agents to act in opposite to their personal security preference [18].
- p_3: Norms Influence - Social proof, where individuals assume the actions of those they have observed in order to reflect the interpreted cultural norms is apparent in many societies [15]. We capture this cumulatively as discussed in Section 3.3.
- p_4: Messenger Influence - *Authority*, sometimes referred to as *Messenger* is influencing by social/professional status, those we perceive to be in a position of power or responsibility can influence our behaviours [8,7].
- p_5: Personality - Different personalities can react differently to the same influences. We implement a notion of a personality trait for agents: *Conscientiousness* - influenced by *Messenger*; *Agreeableness* - influenced by *Social Norms* and *Messenger*; *Extraversion* - influenced by *Social Norms* [20].

For a security practitioner, there is a distinct set of actions and observations that we can record for agents. The security practitioner can record agents moving between locations, tailgating being permitted or denied, agents successfully tailgating, agents fail to tailgate and agents observing tailgating being permitted or denied. This is all public information, the private information, such as some of the parameters is hidden from a security practitioner. They do not know if an agent can be influenced by *Messenger* or *Social Norms*.

5 Assessment Methodology

We associate a model with a set of parameters, we then run a simulation and return a trace. We use cross validation to calculate six properties of interest for a model which are the average error rate, average precision rate, average recall rate standard deviation, the number of agents correctly predicted once and the number of agents correctly predicted twice. In the cross validation, a decision tree is trained with 80% of the data and remaining 20% is used for testing.

A prediction from a decision tree is either *usable* or *secure*. If we consider secure as our target value then a *true-positive* (tp) is a correct prediction for *secure*, *true-negative* (tn) is a correct prediction for *usable*. *False-positive* (fp) is an incorrect prediction for *usable* and *false-negative* (fn) is an incorrect prediction for *secure*. From these we can calculate the error rate, precision and recall:

$$error = \frac{fp + fn}{tp + tn + fp + fn} \quad precision = \frac{tp}{tp + fp} \quad recall = \frac{tp}{tp + fn}$$

The cross validation creates a number of decision trees for each parameterised model. From the cross validation we calculate the required properties.

5.1 Feature Extraction

In order to build a decision tree, we must first extract features from the synthetic data. By features, we mean classes for the training data that a decision tree will use to generate rules. The classes are used in the testing phase to reach a decision. There are many ways to extrapolate features from a dataset. Features such as the number of permits and denies for tailgating is of interest. Additionally, the number of observations for permits and denies is relevant. As we designed the model, we know that observations can influence behaviour, therefore, we separate observations over the five days that the simulation runs for.

1. Bob, t, move, $front$, $inside$, NA
2. Alice, t', move, $outside$, $back$, NA
3. Alice, t'', tailgate, $back$, $inside$, Charlie
4. Charlie, t'', permit, $front$, $inside$, Alice
5. Dan, t'', obsPermit, $front$, $front$, Alice; Charlie

Fig. 4: An example trace with four entries.

In Figure 4 we see a typical trace which follows the format *agent, time, action, start location, end location, agents involved*. The simulation of the models generates a trace similar to Figure 4 but with thousands of entries. From this we must identify the important features. In the case of Figure 4 we know that *Alice* has tailgated, Charlie has permitted it and *Dan* has observed it, however, we have no relevant information for *Bob*, who moved too early into the building and did not engage or observe any relevant actions.

From an assessment perspective, depending upon the underlying rules of the model, i.e. the parameters, a decision tree could predict the security preference of *Alice*, *Charlie* and perhaps *Dan*, as for *Bob*, this is a clear limitation of this method, where decision trees have little power. This is the private information impacting the uncertainty of classifier models, a security practitioner can only record what they witness.

6 Analysis - Case Study

In this section we discuss the use of parameterised models and make remarks surrounding the results for three different cases.

The number of possible parameterised models is 2^5, we only consider 11 of these 32. The expected arrival rate is included in the majority of the parameterised models, as we do not consider too many models where all agents always arrive at the exact same time, of course this could happen, but it is very unlikely. The personality parameter is dependent upon a behaviour change parameter being present, therefore, it does not add to a model if *Social Norms* and/or the *Messenger* parameters are not included.

We used the *Julia* programming language to implement our case study and made use of the SysModels package [6,19]. We generated the synthetic data on a

Toshiba laptop with a 2.4 GHz i5 processor and 8GB RAM. To generate the data for 11 models with 200 agents it took 22 minutes which is roughly 2 minutes per model. Each model is generated with 10 traces each starting from an identical initial state for each model.

For the analysis we performed four test cases and used 50, 100, 150 and 200 agents. Table 5 is the results for the 100 agents, the results for 50, 150 and 200 agents are in the Appendix in Tables 6, 7 and 8 respectively.

p_1	p_2	p_3	p_4	p_5	**Model**	$\mu(error)$	$\sigma(error)$	$pr(s)$	$r(s)$	$n=1$	$n=2$
					m_1	0.255	0.067	0.659	0.830	93	55
✓					m_2	0.001	0.002	1	0.997	100	99
✓	✓				m_3	0.234	0.028	0.697	0.712	94	59
✓		✓			m_4	0.073	0.019	0.963	0.953	99	86
✓			✓		m_5	0.160	0.024	0.884	0.898	96	71
✓		✓		✓	m_6	0.094	0.018	0.928	0.938	98	82
✓			✓	✓	m_7	0.114	0.016	0.904	0.910	98	79
✓	✓	✓			m_8	0.271	0.024	0.724	0.731	92	53
✓	✓		✓		m_9	0.367	0.031	0.634	0.624	86	40
✓		✓	✓	✓	m_{10}	0.027	0.012	0.975	0.969	99	94
✓	✓	✓	✓	✓	m_{11}	0.277	0.028	0.675	0.675	92	52

Table 5: 100 Agents: p_1: Expected Arrival Rate; p_2: Probabilistic Decision; p_3: Norms Influence (Social Proof); p_4: Messenger Influence; p_5: Personality; $\mu(error)$: Average error rate of a model; $pr(s)$: The precision of the model towards *secure*; $r(s)$: The recall of the model for *secure*; $n = 1$: The average times an agent is predicted once with the test data; $n = 2$: The average times an agent is predicted once with the test data;

For each test case, we calculated the average error rate, the standard deviation, the precision and the recall of each parameterised model, where Table 5 shows the parameters for each model. We now make remarks regarding the results we have obtained.

Remark 1 *The average error rate for model m_1 is significantly more accurate with 50 than 100, 150 or 200 agents.*

With regards to Remark 1, as the expected arrival time is not set, all agents arrive at the same time. Due to the conditional rule that an agent will tailgate if less than 5 people are in the queue, this results in a number of agents never actively being involved in a security incident. The majority of agents don't ever permit, deny or attempt to tailgate, therefore, a decision tree will make inaccurate predictions for some agents, particularly when more than 50 agents are used.

Remark 2 *If the probabilistic parameter is set, then the average error rate significantly increases. In particular, it impacts more than both the Messenger and Social Norms parameters.*

The use of the probabilistic parameter significantly increases the average error rate of the decision trees. Due to the uncertainty of agent behaviour, i.e. *secure* agents acting *usable* and vice versa, a *secure* agent could have always behaved as *usable*. A classifier model would always conclude that they are *usable* when they are in fact *secure*. Whilst Remark 2 is not surprising, the impact of uncertain behaviour against social influences is a useful result for a security practitioner. In the real world, some people will always be *secure* or *usable*, some hover between the two and some may slightly more *secure* or slightly more *usable*, having some insight into these numbers would allow us to calculate the impact of agents towards a model based on knowledge that uncertain behaviour reduces the accuracy of classifier models.

Remark 3 *The Messenger influence has a slightly more of an impact to the error rate, precision and recall of a model than Social Norms. This is true for all four of the test cases. More importantly, they both impact the error rate, precision and recall of every model.*

The influences themselves differ in how they are implemented. The *Messenger* relies on an agent observing a behaviour of another agent that they consider to be an authoritative figure. The *Social Norms* is a cumulative influence, where the number of observations of a particular action can trigger the security preference of an agent to change. For Remark 3, the interest is that they are not probabilistic behaviours, they are private behaviours.

The internal context of an agent, their personality, who is authoritative to them and whether or not they are influenced by *Social Proof* is the uncertainty that decision trees struggle to capture. Two agents with the features for security incidents and observations could differ in security preference due to their internal context, how they perceive other agents. Again, this reflects the real world where some people can be easily influenced or socially engineered against, and some people cannot.

Remark 4 *As the number of agents that are on average correctly identified once decreases, the number of agents that are on average correctly identified twice significantly decreases.*

There is a strong correlation between the number of agents identified once and only once against the subsequent number of agents that are identified twice. For example, consider m_2, which is the most accurate model in all areas. For all four test cases, more than 99% of agents are correctly identified once and more than 98% of agents are correctly identified twice. Considering Remark 4, for m_2 the accuracy of the classifier models is clear, there is no behaviour change present and agents stochastically arrive to the workplace. We expect that the distribution of security incidents for agents is evenly distributed, as such, an accurate decision tree can is trained.

When we consider the probabilistic models where the number of agents correctly identified once is significantly lower, less than 90% in some cases. The number of agents identified twice drops significantly, indicating that the traces

generated for these model have such a large variation in features, a classifier struggles to predict an agents security preference.

Remark 5 *On average, the models for 200 agents are more accurate than 50, 150 and 100 agents.*

A trend emerged for the accuracy of models as we increased the number of agents. Whilst some of the models were more accurate for 50 agents, in general Remark 5 holds, in particular for the complex models where influences and probabilistic decisions are present. This is due to an increase number of entries to train decision trees, improving its accuracy.

7 Conclusion

In this paper we designed a multi agent system to generate synthetic data. Secondly, we identified features from the synthetic data. Finally, using cross validation we trained and tested many different decision trees for four test cases.

The generated decision trees showed that as the complexity of human behaviour increases, the less accurate decision trees are for predicting attributes, in this case, the security preference of employees. The remarks from Section 6 highlight the important features of the models with regards to the parameters. For example, probabilistic decisions impact the model significantly more than influences with regards to the error rate, precision and recall. Between the influences, the *Messenger* influence had a greater impact, however, this is partially down to how a security practitioner or designer implements behaviour change.

For a security practitioner, the insights towards the impact of these different parameters allows for an understanding between the limitations of decision trees and predicting security preferences. In particular, the certainty one can put in the accuracy of the decision trees.

The future of this work will target the unanswered questions that we can draw from this paper. Calculating the impact the parameters have towards error rate, precision and recall would allow a security practitioner to identify when probabilistic agents, influences or any other behaviours are present without having prior knowledge as we did. The techniques for building classifier models will be explored, for example, by considering different algorithms for building classifier trees, or sampling a range of features to assess the importance of each feature.

References

1. A. Beautement, M. A. Sasse, and M. Wonham. The compliance budget: managing security behaviour in organisations. In *Proceedings of the 2008 workshop on New security paradigms*, pages 47–58. ACM, 2009.
2. B. J. Biddle. Recent developments in role theory. *Annual review of sociology*, 12(1):67–92, 1986.

3. J. Blythe, R. Koppel, and S. W. Smith. Circumvention of security: Good users do bad things. *IEEE Security & Privacy*, 11(5):80–83, 2013.
4. T. Caufield and S. Parkin. Case study: Predicting the impact of a physical access control intervention. In *STAST (Socio-Technical Aspects of Security and Trust)*. In publication., 2016.
5. T. Caulfield, M. Baddeley, and D. Pym. Social learning in systems security modelling. *constructions*, 14(15):3.
6. T. Caulfield and D. Pym. Improving security policy decisions with models. *IEEE Security & Privacy*, 13(5):34–41, 2015.
7. R. B. Cialdini and N. Garde. *Influence*, volume 3. A. Michel, 1987.
8. P. Dolan, M. Hallsworth, D. Halpern, D. King, R. Metcalfe, and I. Vlaev. Influencing behaviour: The mindspace way. *Journal of Economic Psychology*, 33(1):264–277, 2012.
9. R. H. Frank. If homo economicus could choose his own utility function, would he want one with a conscience? *The American Economic Review*, pages 593–604, 1987.
10. H. H. Kelley. Attribution theory in social psychology. In *Nebraska symposium on motivation*. University of Nebraska Press, 1967.
11. R. K. Merton. *Social theory and social structure*. Simon and Schuster, 1968.
12. S. Michie, M. M. van Stralen, and R. West. The behaviour change wheel: a new method for characterising and designing behaviour change interventions. *Implementation science*, 6(1):42, 2011.
13. C. Morisset, I. Yevseyeva, T. Groß, and A. van Moorsel. A formal model for soft enforcement: influencing the decision-maker. In *International Workshop on Security and Trust Management*, pages 113–128. Springer International Publishing, 2014.
14. A. S. Rao and M. P. Georgeff. Modeling rational agents within a bdi-architecture. *KR*, 91:473–484, 1991.
15. H. Rao, H. R. Greve, and G. F. Davis. Fool's gold: Social proof in the initiation and abandonment of coverage by wall street analysts. *Administrative Science Quarterly*, 46(3):502–526, 2001.
16. M. Schlüter, A. Baeza, G. Dressler, K. Frank, J. Groeneveld, W. Jager, M. A. Janssen, R. R. McAllister, B. Müller, K. Orach, et al. A framework for mapping and comparing behavioural theories in models of social-ecological systems. *Ecological Economics*, 131:21–35, 2017.
17. R. S. Shaw, C. C. Chen, A. L. Harris, and H.-J. Huang. The impact of information richness on information security awareness training effectiveness. *Computers & Education*, 52(1):92–100, 2009.
18. R. H. Thaler. From homo economicus to homo sapiens. *The Journal of Economic Perspectives*, 14(1):133–141, 2000.
19. Tristanc. Sysmodels package. `https://github.com/tristanc/SysModels`, Feb 2017. [Online; accessed 08-June-2017].
20. S. Uebelacker and S. Quiel. The social engineering personality framework. In *Socio-Technical Aspects in Security and Trust (STAST), 2014 Workshop on*, pages 24–30. IEEE, 2014.
21. I. H. Witten, E. Frank, M. A. Hall, and C. J. Pal. *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann, 2016.
22. M. Wooldridge. *An introduction to multiagent systems*. John Wiley & Sons, 2009.
23. F. Zhu, S. Carpenter, A. Kulkarni, and S. Kolimi. Reciprocity attacks. In *Proceedings of the Seventh Symposium on Usable Privacy and Security*, page 9. ACM, 2011.

Appendix A Additional Results

Model	$\mu(error)$	$\sigma(error)$	$pr(s)$	$r(s)$	$n=1$	$n=2$
m_1	0.070	0.037	0.896	0.943	49	43
m_2	0.018	0.010	0.974	0.980	50	48
m_3	0.279	0.035	0.658	0.642	45	26
m_4	0.050	0.025	0.947	0.950	49	45
m_5	0.162	0.029	0.853	0.867	48	35
m_6	0.031	0.018	0.955	0.977	49	47
m_7	0.091	0.030	0.893	0.937	49	41
m_8	0.266	0.039	0.701	0.686	46	27
m_9	0.365	0.051	0.694	0.656	43	20
m_{10}	0.017	0.012	0.976	0.986	50	48
m_{11}	0.325	0.056	0.622	0.581	44	23

Table 6: 50 Agents: p_1: Expected Arrival Rate; p_2: Probabilistic Decision; p_3: Norms Influence (Social Proof); p_4: Messenger Influence; p_5: Personality; $\mu(error)$: Average error rate of a model; $pr(s)$: The precision of the model towards *secure*; $r(s)$: The recall of the model for *secure*; $n=1$: The average times an agent is predicted once with the test data; $n=2$: The average times an agent is predicted once with the test data;

Model	$\mu(error)$	$\sigma(error)$	$pr(s)$	$r(s)$	$n=1$	$n=2$
m_1	0.308	0.043	0.677	0.644	132	74
m_2	0.001	0.002	0.999	0.998	150	149
m_3	0.268	0.021	0.656	0.675	139	80
m_4	0.088	0.020	0.948	0.951	148	125
m_5	0.243	0.026	0.814	0.814	140	86
m_6	0.132	0.015	0.906	0.919	145	114
m_7	0.163	0.021	0.881	0.883	144	106
m_8	0.275	0.029	0.713	0.715	138	78
m_9	0.375	0.027	0.635	0.617	129	58
m_{10}	0.042	0.013	0.949	0.966	149	137
m_{11}	0.259	0.027	0.682	0.685	139	82

Table 7: 150 Agents: p_1: Expected Arrival Rate; p_2: Probabilistic Decision; p_3: Norms Influence (Social Proof); p_4: Messenger Influence; p_5: Personality; $\mu(error)$: Average error rate of a model; $pr(s)$: The precision of the model towards *secure*; $r(s)$: The recall of the model for *secure*; $n=1$: The average times an agent is predicted once with the test data; $n=2$: The average times an agent is predicted once with the test data;

Model	$\mu(error)$	$\sigma(error)$	$pr(s)$	$r(s)$	$n=1$	$n=2$
m_1	0.201	0.135	0.861	0.912	177	142
m_2	0.006	0.003	0.996	0.998	199	197
m_3	0.170	0.013	0.910	0.888	190	141
m_4	0.014	0.006	0.993	0.993	199	194
m_5	0.050	0.009	0.976	0.972	196	183
m_6	0.024	0.007	0.984	0.990	199	191
m_7	0.047	0.011	0.976	0.973	198	182
m_8	0.140	0.021	0.933	0.912	192	151
m_9	0.277	0.029	0.833	0.812	183	105
m_{10}	0.040	0.008	0.975	0.980	197	186
m_{11}	0.161	0.016	0.920	0.892	191	144

Table 8: 200 Agents: p_1: Expected Arrival Rate; p_2: Probabilistic Decision; p_3: Norms Influence (Social Proof); p_4: Messenger Influence; p_5: Personality; $\mu(error)$: Average error rate of a model; $pr(s)$: The precision of the model towards *secure*; $r(s)$: The recall of the model for *secure*; $n=1$: The average times an agent is predicted once with the test data; $n=2$: The average times an agent is predicted once with the test data;