Spring 8-2013

User Modeling via Machine Learning and Rule-Based Reasoning to Understand and Predict Errors in Survey Systems

Leonard Cleve Stuart
University of Nebraska-Lincoln, leonard.stuart@huskers.unl.edu

USER MODELING VIA MACHINE LEARNING AND RULE-BASED REASONING TO UNDERSTAND AND PREDICT ERRORS IN SURVEY SYSTEMS

By

Leonard Cleve Stuart

A Thesis

Presented to the Faculty of

The Graduate College at the University of Nebraska

In Partial Fulfillment of Requirements

For the Degree of Master Science

Major: Computer Science

Under the Supervision of Professor LeenKiat Soh

Lincoln, Nebraska

August, 2013

USER MODELING VIA MACHINE LEARNING AND RULE-BASED REASONING TO UNDERSTAND AND PREDICT ERRORS IN SURVEY SYSTEMS

Leonard Cleve Stuart, M.S.

University of Nebraska, 2013

Advisor: LeenKiat Soh

User modeling is traditionally applied to systems were users have a large degree of control over their goals, the content they view, and the manner in which they navigate through the system. These systems aim to both recommend useful goals to users and to assist them in achieving perceived goals. Systems such as online or telephone surveys are different in that users have only a singular goal of survey completion, extremely limited control over navigation, and content is restricted to prescribed set of survey tasks; changing the user modeling problem to one in which the best means of assisting users is to identify rare-actions hazardous to their singular goal, by observing their interactions with common contexts. With this goal in mind, predictive mechanisms based on a combination of Machine Learning classifiers and survey domain knowledge encapsulated in sets of rules are developed that utilize user behavioral, demographic, and survey state data in order to predict when user actions leading to irreparable harm to the user's singular goal of successful survey completion will occur. We show that despite a large class imbalance problem associated with detecting these actions and their associated users, we are able to predict such actions at a rate better than random guessing and that the application of domain knowledge via rule-sets improves performance further. We also identify traits of surveys and users that are associated with rare-action incidence. For

future work, it is recommended that existence of potential sub-concepts related to users who perform these rare-actions be explored, as well as exploring alternative means of identifying such users, and that system adaptations be developed that can prevent users from performing these rare and harmful actions.

Dedicated to Banjo Hoolihan, dog.

Acknowledgements:

This thesis is part of a larger team project at the University of Nebraska – Lincoln (UNL), and thus would not have been possible without the opportunity to work closely with multiple students and faculty both in the Intelligent Agents and Multiagent Systems (IAMAS) group of the Computer Science department and the Survey Research and Methodology (SRAM) group here at UNL. Specifically we would like to thank Gregory Atkin, Adam Eck, LD Miller, Clayton Brady, and Dr. LeenKiat Soh of the IAMAS group for helping to form ideas, software, and databases utilized in this thesis and for offering advice and guidance. And to thank Tarek Baghal, Dr. Robert Belli, Dr. Alan McCutcheon, Lynn Phillips, Nick Ruther, Rebecca Powell, Ana Lucia Cordova-Cazar, and Dato Tsabutashvili for lending their invaluable insights into the survey domain, and their wealth of knowledge regarding all aspects of the two survey studied in this thesis. Without their knowledge, guidance, and expertise it would not have been possible to develop the systems used in this thesis, nor to pursue the goals of this thesis. Their contribution cannot be overstated.

This material is based upon work supported by the National Science Foundation under Grant No. SES-1132015.

Table of Contents

Table of Figures

CHAPTER 1

INTRODUCTION

Computer user modeling is the process by which information is gathered about users, and used to adapt the underlying system to the needs of users (Kobsa, 2001). User modeling is useful in systems where complex, widely available software can be improved by catering to individual user needs and interests (Fischer, 2001). It has been traditionally applied to systems that have the following traits: a high degree of user control over content navigation, a large amount of user self-direction in terms of goals, and a systematic design with usability in mind; furthermore it is applied to assist in the most-common forms of user actions. For example, recommendation systems employ user models to suggest products to the user. However, these systems are part of larger commercial applications centered around user browsing for content or products of their choosing. Furthermore, these systems are thoroughly designed with ease of use in mind (usability). Examples include Netflix, Amazon.com, and other such commercial websites

(Shani and Gunawardana, 2011). However, there exist systems such as online and telephone computerized surveys that do not have such traits. In these systems, user control over content navigation is low, goals are system directed, and usability is limited by intrinsic traits of the application. How would these differences in key system traits impact user modeling? To date, no work has been conducted to specifically address this type of system. As survey systems and survey-like systems are getting more prevalent and ubiquitous in our increasingly digital world, it is important to investigate, determine, and even identify user modeling approaches that would work for such a system.

This thesis designs user models for two examples of such a system, namely, online and telephone surveys. Surveys represent this grouping as content navigation is dominated by the questions found in the survey, goals for survey users are limited to the goal of survey completion, and intrinsic features of surveys dissuade users from quality survey completion, impeding usability. Further note that due to the restricted goal set, the types of assistance user modeling can offer is intrinsically different. In fact the constructed models are used to predict the appearance of survey errors, which represent a rare form of user actions. Hence, our models are built for an environment distinct from most user modeling problems and addressing rare-actions in contrast to most applications addressing common-actions. For shorthand, throughout the thesis we will refer to this modeling concept (entailing the environment and action differences) using the terms rare-action, non-rare action, rare-action users, and non-rare action users. The user models are based on machine learning techniques looking for actionable, operational patterns that (1) could help design better surveys to prevent or reduce the errors, or (2) could allow a self-

administered, agent-powered survey system to recognize and mitigate such rare-actions at real-time.

Before commencing a further discussion on the user modeling problem, it will be helpful to provide a context for the application domain of surveys.

1.1. Survey Background

All surveys have a common problem of non-response and measurement error. Non-response error is user failure to answer either a whole survey or part of a survey (Bosnjak and Tuten, 2001). Measurement error is defined as the recording of inaccurate response data (Couper, 2000). These errors occur, for example, in surveys such as the American Time Use Survey (ATUS) and the Gallup panel, both of which are specifically addressed with the user models in this study.

The American Time Use Survey (ATUS) is a survey conducted by the Bureau of Labor Statistics (BLS) that gathers information on "how, where, and with whom Americans spend their time" (ATUS overview page: http://www.bls.gov/tus/overview.htm). An interviewer conducts the interview over the phone and records responses into a software instrument. ATUS also gathers demographic traits about respondents, and captures how the interviewers interact with a software instrument used to record interview responses. These interactions are captured in log files called audit trails or paradata, and can be thought of as encapsulating the “process data” of the interview (Heerwegh, 2003). That is, audit trails record when an entry was recorded, when it was rewritten, the order items were recorded in, how items were recorded, etc. With this data one can reconstruct the interview process.

In terms of response data, ATUS records what activities a user engages in throughout one whole, single, day by recording a time use diary. A time use diary is essentially a sequential record of a respondent's activities ordered by the time at which each activity occurred. According to the Survey Research and Methodology (SRAM) group at the University of Nebraska – Lincoln (UNL), the sequential nature of responses has implications regarding how previous activities may affect recall (ability to remember) and thus accuracy of subsequent activities. Such records have the potential to be used in recall failure error prediction, a form of non-response error.

The Gallup panel records opinion data from a pre-recruited, recurring panel of respondents. These panel respondents are offered one survey a month (at most) to complete. Each survey is delivered via the Internet and consists of a series of pages with questions on each page. Upon the failure to interact completely with six straight surveys they are removed from the panel. This action is called "attrition". Furthermore, respondents can begin a survey but fail to complete it, which is known as "breakoff". Both of these errors can be classified as non-response errors. The breakoff error is the rare-action of interest to our Gallup-centered user-models.

A third form of non-response is when a respondent submits a survey but fails to answer a given question, which is possible in Gallup. Note that this type of non-response error is trivial to observe. Gallup also records paradata, which captures the order in which respondents answered questions, how long they viewed questions, etc.

In ATUS, the rare-action of interest for this study is related to item non-response error and measurement error. Item non-response errors are a subclass of non-response error that can occur when a user does not remember an action and thus does not complete

one single part of the survey (i.e. a question or single activity) (American Time Use Survey (ATUS) Coding Rules 2008). This could also be thought of as a measurement error, as the total data quality of ATUS is impeded when user's do not recall an activity. This is the rare-action of interest for the models presented in this thesis. Other measurement errors we are interested in dealing with in the future for ATUS include identifying interviews of poor quality, which are currently determined by subjective flags issued by the interviewers, and predicting when a user will refuse to share information.

To summarize, this study uses user models to address rare-action breakoff error in the Gallup panel. Again, breakoff is again a form of non-response error which causes the loss of response data for the entire interview; it is simply a user quitting a survey before having finished. It is difficult to predict, commonplace in web surveys, and is thought to be a result of a complex combination of survey page features, question features, and respondent features (Peytchev, 2009). It is thus a difficult and important error to address in web surveys. In ATUS, the rare-action of interest is memory failures by respondents. Predicting memory failures would be of value to survey methodologists and psychologists capable of developing methods to tease information from users. We could team with such experts to build intelligent systems capable of asking for information in a manner better capable of extracting information from such users, and doing so before the user encounters the memory failure; hence reducing their frustration and improving their experience without having to take mitigating steps after the fact. Like breakoff, memory failure is not easy to predict but is thought to be related certain factors such as age.

1.2. User Modeling

User modeling has been traditionally applied to applications where the user has a set of self-determined goals they are trying to accomplish. A large role of the model is to infer such goals and assist the user in accomplishing them (Biswas and Robinson, 2010). In contrast, "survey respondents are often not deeply vested in the accuracy of their answers" and "their primary goal is likely to involve finishing the interview" (Conrad, Schober, and Coiner 2007). In other words, respondents are not self-driven in surveys to achieve sets of goals they have defined themselves. Instead, they are pitted against a series of system-selected tasks that they have no say in determining, with the singular goal being survey completion. This distorts the role of a user model from detection and assistance in goal achievement, towards efforts to keep the user motivated at achieving a singular, external goal. One way to do this is to have the model address actions that are damaging towards the singular goal; such actions are often rare in the surveys we study.

Similarly, survey software restricts navigation through the application's content to pre-defined sets of questions. In traditional applications with user modeling, users have a large amount of leeway in determining what parts of the application they are interacting with and a major role of user modeling is inferring intended navigation, or advising the user towards beneficial navigation (Brusilovsky, 2001). For example, modern news sites suggest users browse certain stories based on their modeled interests. In contrast, applications such as surveys limit the personal assistance they can offer as they are forced to present the same content to all users, no matter how different each user may be.

Furthermore, restricted navigation forces the user to confront content they may find unappealing. In the domain of surveys, this translates to questions the user finds difficult, topics the user has no interest in, or the length of the survey itself (Peytchev,

2009). Survey methodologists acknowledge that these and other features of the survey can deter users from completing the survey completely, or accurately due to cognitive issues (Krosnick, 1991). Thus, the restricted content the application presents is itself hostile to the goal of survey completion and accuracy. This trait represents another handicap an application like surveys has compared to traditional user modeling application domains.

For example, hypermedia systems equipped with user models for application adaptation include - "educational hypermedia, on-line information systems, on-line help systems, information retrieval hypermedia, institutional hypermedia, and systems for managing personalized views in information spaces" have as their overall design aim assistance in helping users achieve goals (Brusilovsky, 2001). Note that this design aim is applied to the entire structure of the user interface – it is not just expressed through the user model. Hence, features deemed distracting from user goals are eliminated, or are adapted to a more useful setting. In applications where content cannot be eliminated, this option is no longer on the table.

Thus, although survey methodologists do design for ease of completion in mind, they cannot avoid the difficulty of the survey task itself, and that certain survey features are intrinsically *hostile* to certain users. These difficulties thus introduce a new challenge to user modeling.

In summary, applications in which (1) goals are not self-directed (**lack of self-direction**), (2) user control over content navigation is limited (**limited content navigation**), and (3) content is restricted (**restricted content**), represent a new challenge to building user modeling capable of assisting users. Surveys represent a domain where

these features are present, and this work builds models capable of assisting users by predicting the occurrence of survey error. Since the errors we predict are rare we refer to this overall problem as the rare-action user modeling problem.

The survey errors these models predict can be categorized amongst three general scopes:

- Application level error – defined as the user no longer participating in the survey
- Session level error – defined as the user quitting the survey in mid-session
- Action level error – defined as the user making a mistake when performing a small action, such as accidentally selecting the wrong answer choice.

These scopes can be applied to hypermedia systems—not just surveys—in general, as users maintain a history of interaction with a given application (related to application level error), have a set of goals they are attempting to achieve in any given use session (and thus risk session level errors), and within a session attempt to achieve many small goals or tasks (and thus risk action level errors).

Specifically, in this thesis user models were developed that predict survey breakoff in the Gallup panel which can be defined as the decision to no longer continue in a survey (Peytchev, 2009). This is a form of session level error. In ATUS, user models were developed capable of predicting session level error, namely memory recall failure. We categorize this as session level as it represents a failure of the user to complete a whole section of the survey, thus affecting the entire state, and is not simply an isolated action-error such as selecting the wrong answer choice.

1.3. Motivation

From a research level perspective, user modeling of surveys confers numerous benefits:

1. It expands the field of user modeling and adaptive design to a challenging application domain in which 1) **user goals are not self-directed**, 2) **user control over content navigation is limited**, 3) **content is restricted**, and 4) **the actions addressed by the model are rare rather than common**. These features limit the range of observations that can be made about individual users and, as a consequence, ultimately limit the breadth of adaptation available to the system. Developing successful models in this application domain allows researchers to better pinpoint potential adaptations, or can inform system redesign if the desired adaptation is restricted by the application's inherent constraints.
2. Specifically, it lays the groundwork for the construction of a self-administered survey equipped with an intelligent agent able to assist users in the survey task. Constructing such a survey will provide a software architecture allowing survey methodologists a more economical means to conduct surveys; and will come complete with error mitigation technology, enhancing data quality. The first task in this endeavor is the challenge user modeling as described in item 1 above; and hence this thesis is part of this larger project.
3. To build successful models, large amounts of data must be compiled, understood, processed. This task, as a consequence, provides the survey research community with a large database for further study. It further provides a framework by which data can understood and processed. Thirdly, the processed data provides another database by which survey methodologists can construct models based on more abstract data and concepts.

From an application specific and survey informatics perspective, an intelligent user model will empower self-administered surveys to help users avoid frustration and mitigate error. It can be a valuable part of a larger interface designed to be helpful, easy to use, and capable of providing timely feedback such as that presented by Conrad et al (2007).

Furthermore, models or another intelligent system can automatically gather information about traits of interviews or surveys associated with error or frustration and inform survey redesign.

1.4. State of the Art

Related work and background will be discussed further in Chapter 2, but here we present a brief overview of user modeling and adaptive survey design.

User modeling has long been applied to predicting user actions in interfaces. Such work includes the Lumiere project (Horvitz et al, 1998), which was notably groundbreaking and flawed as end-users ultimately found the intelligent assistant irritating. The Lumiere project used Bayesian techniques to make predictions.

Other works leveraging Bayesian techniques are by Chen et al (2002) and Pardos et al (2010). Chen et al was able to predict future user web browsing actions by using previous data; and Pardos et al was able to predict student mistakes in a tutoring system by using previous data.

One could also view survey error as a form of anomalous behavior departing from the typical survey case. From this perspective, Salem and Stolfo (2011) were able to detect security breaches by finding anomalous behavior using support vector machines. Support vector machines can also work with sequential data in other ways given an

appropriate kernel. Thus both anomaly detection and SVMs offer potential avenues to predicting survey error.

An interesting aspect of the Gallup panel is the recurring participation of various users. Thus while building global error predicting models might be useful, it would be further valuable to exploit known individual traits about users. Pardos and Heffernan (2010) used such techniques to individualize a more global intelligent tutoring assistant based on previous individual student interactions. Similar work by Manavoglu et al (2003) leveraged maximum entropy models and hidden Markov models in the prediction of user behavior on Cite Seer. They pointed out that this had the advantage of both addressing the cold start problem and allowing the exploit of user data. In this work we focus on global models and leave further individual refinement for future work.

Engelbrech et al (2009) also used hidden Markov models in the prediction of user frustration during an automated telephone interaction (e.g. something similar to the process one goes through when paying the cable bill over the phone). This work is very interesting from the perspective of predicting interview error, as some theoretical causes of survey error are related to mental weariness, and hence frustration (Krosnick, 1991).

User models capable of predicting user behavior or interests have also been built using clustering methods (Virvou et al, 2012), and by evaluating users' value systems (Lakiotaki et al, 2011). One drawback of this latter approach is that the "values" must be created *a priori*. However, the work is interesting as it is analogous to if we view survey users as possessing competing goals to finish or quit a survey.

Finally, case-based reasoning has been applied to user action prediction (Armentano et al, 2012; Cordier et al, 2010). One interesting aspect of the work by

Armentano et al, is that it models user actions based on “plans” and “goals”; associating different plans (i.e. user actions) with goals. The work by Cordier et al is notable for its application of the same semantic meaning to disparate user actions such as a user applying two different mechanisms to copy and paste. This is enticing, as it presents a means of associating different user sequences with the same ultimate outcome. However, in our domain of predicting survey error creating such a catalogue of plans and goals depends on knowing exact mechanisms by which memory recall failure or breakoff might arise, a problem currently not in reach.

Conrad et al (2007) broke ground in their work, by building the ability of users to garner “clarification dialogue” to complicated, factual questions into a laboratory based web survey. They further broke ground by providing this text either on demand, or when a user-model indicated clarification was needed. Clarification was triggered based on how long the user had been reading the question. Two kinds of user models were employed, a generic user model which had a common clarification time limit, and a set of stereotyped user models which varied the time limit based on respondent age. They compared the use of model-based clarification with user requested clarifications and a system that always provided clarification. They were able to show that clarification text, and specifically text triggered by time limits was successful in their study, though they cautioned that the survey took place in a laboratory setting. This work is important as it proves that the concept of intelligent survey assistance is feasible and potentially valuable, and thus it is worth the trial of extending the work.

1.5. Proposed Solution

The goal of this work is to build effective user models in a challenging application domain. It is argued that user modeling for surveys contrasts itself to typical user modeling problems because

1. Applications such as surveys are **content restricted**. Surveys are a potentially hostile environment for an end user. Survey methodologists agree that user interest may be lacking from the get-go and may be made worse from the nature of the survey itself, i.e., its content. This is very different from typical applications employing user modeling, where the entire application is designed to assist the user achieve a goal and the model is a natural extension of this design. Here, content is designed to be accommodating or flexible.
2. Most user modeling is aimed at detecting certain micro-goals to proffer intelligent assistance (i.e., suggesting papers for users conducting a search). For surveys, the goal of user modeling is to detect the user straying from a singular goal common to all users of completing the survey (also a system goal). This is different from typical goal straying because the users are not capable of simply drifting to a new, distorted goal, but instead can either successfully achieve the singular goal or not. It is claimed that applications such as surveys are thus characterized by a **lack of self-direction** in terms of goals, to some extent fundamentally changing the user modeling problem.
3. Users do not direct the series of action, or content navigation, in surveys. Instead they are offered a series of prescribed tasks to complete. In the traditional user modeling problem, the environment is open-ended and up to the user to determine the navigation path. In fact, many user models attempt to help users find a path they would find uniquely interesting or useful. This option is not available to survey users.

Instead the goal of user modeling is to ensure that users do not find the prescribed path too troublesome. This **limited content navigation** makes effective user modeling and interface adaptation more difficult.

To address these challenges, effective user modeling strategies must be developed. The strategies developed in this thesis build models aimed at detecting the presence of rare-actions destructive to the user's singular goal, hence addressing the problem by offering the system knowledge of user trouble before it occurs, allowing for preventive, adaptive measures to be taken. These models are built to function in an environment possessing content-restriction, limited content-navigation, and possessing a goal set undetermined by users.

The models are based on **machine learning (ML) classifiers equipped with secondary classification filtering technique based on sets of rules derived domain knowledge**. The algorithms also make use of **resampling techniques** often applied to ML problems when the dataset has class imbalance issues. Since the user actions we predict are rare, the datasets we encounter have large imbalance issues. We will discuss the rule-sets and resampling issues further in Chapters 2 and 3.

The ML classifiers employed are the **naïve Bayes**, **artificial neural network** (or Multilayer Perceptron), and **decision tree classifiers**. We chose the decision tree and naïve Bayes classifiers as they require short training times (which could prove valuable in a real-time system) and because the classification structure is interpretable by humans which can provide guidance to future work. We chose artificial neural networks in order to investigate if this classifier can better exploit the numeric data prevalent in the datasets derived for the user modeling problem, and to see if its ability to find complex

relationships yields more fruitful results. In the end, the naïve Bayes and decision tree based techniques seemed to outperform the artificial neural network.

To address the class imbalance, resampling using **oversampling the minority class**, **undersampling the majority class**, the **Balanced Cascade** resampling technique (Liu et al, 2009), and the **Easy Ensemble** resampling technique (Liu et al, 2009) are investigated. The Easy Ensemble and Balanced Cascade techniques both develop an ensemble of classifiers, with each member trained from oversampling the minority class. We also investigate the use of no resampling.

The ML classifiers are augmented with a secondary rule-based technique used to override ML classifications, this was motivated by stagnant performance of the classifiers. The rule-based technique is composed of two rule-sets: the **True Positive Rule Set** and the **True Negative Rule Set**. The role of the True Positive Rule Set is to overturn erroneous ML classifications stating a user is not a rare-actor, and the role of the True Negative Rule Set is to overturn erroneous ML classifications stating the user is a rare-actor. The rule-sets are composed of a set of rules, each of which associates a group of attribute-values with either rare-action or non-rare action. Each rule works by examining an attribute-value and comparing it to a boundary value via a comparison operator. If the result of the comparison is true, then that indicates the rule has flagged the user as belonging to the user-group its rule-set is in charge of detecting. Thus if a member of the True Positive Rule Set returns a true value when examining an attribute-value, this means that the rule is flagging the user as a rare-action user. Each flag is then considered a vote, and if the total votes of all members of the rule-set exceed a set threshold, the user is reclassified into the category the rule-set is in charge of detecting. In our experiments

we compare the performance of detecting rare-action users both with and without the rule-sets and find the rule-sets improve performance.

Again, **the rule-sets are based on domain knowledge**. For example, survey methodologists claim that older respondents are less likely to breakoff from surveys (Peytchev, 2009), and so we can build this knowledge into the rule-sets via rule such as users over age 70 should not be considered rare actors. Developing rules containing domain knowledge has been done in user modeling problems in the past (Frias-Martinez set al, 2006), and we find in our studies that it is effective here. **We have also automated the extraction of rule-sets from user data**.

To reiterate, the goal of the project is to build user models capable of assisting users in a domain characterized by a **lack of self-direction**, **limited content navigation**, and **the presence of restricted content**. Machine learning algorithms were employed and investigated to construct such models, relying on data derived from user interaction with surveys. These classifiers were trained using resampling techniques, in order to address class balance issues, and then equipped with secondary, domain knowledge based, rule-set technique used to override erroneous classifications. We were able to develop models that predict rare-actions (i.e., survey error) in this hostile environment at rate better than randomly guessing. Furthermore, we were able to show that including the rule-set technique improves performance.

Investigations into the most relevant features to be used for modeling revealed that the prevalence of certain types of multiple choice questions are related to the presence of the breakoff rare-action in Gallup, as is the bulk presence of questions and the number of words appearing on a page. It was also found that in some surveys

observing possible measurement errors is associated with rare-actions. Quite significantly we also found that rare-actions tend to happen quite early in the Gallup panel. Each of these findings has important implications for improving the presented methodology used to detect rare-actions that will be discussed in Chapters 5 and 6.

1.6. Contributions

This project makes several contributions:

- It creates user models applicable to an environment characterized by a **lack of self-direction**, **limited content navigation**, and **the presence of restricted content**.
- It demonstrates the feasibility of applying a two-tiered classifier system—based on data-driven ML and domain knowledge-driven rule-based reasoning—to address problems in user modeling.
- It finds features of the environment (i.e., surveys), user behavior, and user traits associated with rare actions, thus providing information for future models. Importantly we find that rare-actions occur early in the Gallup panel and in the presence of certain question types. These findings give much insight into how to improve the system in future work.
- Programs were created that
    - Can recreate user sessions and experience in both Gallup and ATUS.
    - Can build classifiers capable of classifying users are rare-actors or non-rare actors.
    - Can automatically extract domain knowledge from user data and place that knowledge into rule-sets used to improve user classification.

Generally speaking, contributions have been made in the fields of computer science and survey science, and in the practical implementation of user models for ATUS and the Gallup panel.

1.7. Overview

The rest of this Thesis is organized as follows.

Chapter 2 describes related work and background, overviewing related work in the fields of user modeling in Section 2.1; user modeling techniques in Section 2.2; an explanation of why we use user modeling in assisting survey users in Section 2.3; the process of user modeling in Section 2.4; discusses the class imbalance problem in Section 2.5; and closes with a discussion of the theoretical causes of survey error and its implications to the domain based rule-sets in Section 2.6

Chapter 3 describes the methodology. It begins in Section 3.1 the methods by which users are modeled. This includes information on how data is gathered, treated, and how user classification mechanisms are built. Section 3.2 then describes the methodology applied to the feature extraction goals of the thesis. Section 3.3 then illustrates the rule-sets more concretely by describing examples of rules.

Chapter 4 covers implementation. It describes how the data on which the models are based was collected in Section 4.1, how this collected data was preprocessed for the models in Section 4.2, how the user-classification (i.e., predictive mechanisms) for the models were built in Section 4.3, and how the rule-sets were implemented in Section 4.4, and briefly discusses how the classification mechanisms were evaluated in Section 4.5.

Chapter 5 then presents and discusses results of several experiments. We first overview the Chapter, and then Sections 5.1 and 5.2 present information regarding evaluation metrics used to evaluate the experiments. Section 5.3 then shows the results of applying ML classifiers and resampling to the classifying users as rare-actors in the Gallup panel, demonstrating that we can do so at a rate statistically superior to random guessing. Section 5.4 then investigates whether the application of domain knowledge via hand-crafted rules improves classification performance in the Gallup panel, and finds that it does, though with some reservations. Section 5.5 then demonstrates how automated extraction and application of rule-sets as a secondary technique amended to the ML classifiers improves user categorization in the Gallup panel. We argue that this technique is more feasible, generalizable, and experimentally verifiable than the hand-crafted technique. Section 5.6 further explores the use of automated rule extraction and classification by applying the method to the ATUS user modeling problem. We find statistical improvement in this domain as well. Section 5.7 then extracts environmental features in the Gallup panel associated with rare actions. Section 5.8 extracts behavioral and demographic features associated with rare-actions in the ATUS and Gallup panel. Throughout Chapter 5 we observe many interesting patterns in the experimental results and discuss their implications for both the current system and future work.

Finally, Chapter 6 draws conclusions from the current results and overviews ideas regarding future work. Section 6.1 focuses on conclusions while Section 6.2 addresses future work.

Chapter 2

Related Work & Background

In this chapter we cover related work in terms of user modeling. We argue that we apply the user modeling framework to a problem not traditionally addressed by the field – the detection and prevention of rare user actions; and that furthermore our system is different than most user modeling applications in that there is a fixed, singular goal available to users.

Since our models are based on ML algorithms, we also describe basic concepts related to machine learning in the face of imbalanced data. Our data is imbalanced due to the fact that the actions we are attempting to predict our rare, and thus we must apply appropriate resampling techniques when learning, and appropriate analysis techniques when measuring performance. Finally, we discuss the overlapping sources and causes that survey errors have, including the errors associated with the rare actions our models aim to detect. This overlap, and its consequences, has important implications for our

domain-based rule technique, specifically upon the techniques needed to search for ideal rules. We discuss each of these concepts in the following sections.

2.1 Goals of user modeling and Places we see user modeling

Two fundamental terms in user-adaptive systems are the terms adaptable and adaptive. Both terms convey that the system adjusts its presentation or behavior based on traits of the current individual user, but they communicate different mechanisms to achieving this end. In adaptable systems the user can change system settings to their own preferences (Jameson, 2009), whereas in adaptive systems, the system "*adapts its behavior to individual users on the basis of processes of user model acquisition and application that involve some form of learning, inference, or decision making*" (Jameson, 2009). In other words, adaptable systems change their behavior based on explicit user settings, while adaptive systems infer alternative settings or behavior the individual user is likely to find beneficial. **We are interested in building user models capable of powering a system's adaptive capabilities**.

One primary goal of user modeling is to enable adaptation of a system to an individual user (Jameson, 2009). Put another way, the main goal of user modeling is to store information about individual users which can assist the system in "serv[ing] the user better" (Biswas and Robinson, 2010). In other words, the main end of any user modeling system is to improve user experience via personalization. Thus, while other goals differ depending on the domain, the primary thread is to individualize the system in such a way that better fits the user's wants or needs.

Note that user modeling is often applied to situations where users are entertaining a myriad of possibilities about actions to take. In Chapter 1, we described this as users

having self-directed goals in an open-ended navigation and interaction environment. As an example, consider "high-functionality applications" in which "users create new worlds" such as "Unix, MS-Office, and Photoshop" (Fischer, 2001). Such applications have numerous potential user goals, and the role of a model is to identify the current goal of a user and help them achieve it. One example of such an attempt is the Lumiere project described in Chapter 1, in which an assistant was built for Microsoft Office applications that attempted to detect user tasks and offer tips on how to complete them (Horvitz et al, 1998).

Information acquisition systems also employ user models (Jameson, 2009) in order to identify information or sources most relevant to the user (Biroukou et al, 2012; Fischer, 2001). The Implicit system, for example, uses collaborative filtering in order to suggest content to users based on what their colleagues have displayed an interest in (Biroukou et al, 2012). Note the similarity of this task to the high-functionality application problem – there exits a large space of potential end points, i.e., goals in the form of information, a user would like to reach and the role of the model is to help the user reach a point of interest.

Recommendation systems are again very similar to the two previous examples. Recommendation systems suggest products, or items to users based of perceived interest (Jameson, 2009). Examples include Amazon, Netflix, Google Shopping, and many others. These systems too are derived by inferring user goals (or interests) and making suggestions. Furthermore, note that each of theses systems is inherently *open-ended*. That is, in most interactions with the system there is not a steadfast goal the user clings to or is being forced to complete, nor is there a forced path of navigation. Instead, the interaction

is to some extent exploratory, as the user browses news stories or products, and the model can thus help the user to see parts of the system or catalogue they might not encounter, and thus help users simply by exposing possible goals they are likely to be interested in rather than assisting in achieving a singular goal (Lops et al, 2011). Indeed, this is often a desired trait of recommendation systems as it exposes more of the product catalogue to potential buyers. Designers term this feature as "serendipity" (Lops et al, 2011).

A fourth application is tutoring systems (Virvou et al, 2012; Pardos et al, 2006; Pardos and Heffernan 2010; Malpani et al, 2011). The goal of such systems is often to identify what topic the students are struggling with, to lump students into groups, or to determine when to provide certain kinds of information. These systems too are goal-diverse, as users can have very different learning objectives (in the form of topics to learn) or needs. They can also be open-ended if the users are allowed to choose how much practice they receive with certain learning topics, etc.

Another application of user modeling is the automation of routine tasks and adapting spoken dialogue systems to individual users (Jameson, 2009). An example of task automation is sorting email or scheduling appointments, and an example of adaptive dialogue systems modern public transportation ticket counters in which the user is guided through the process of purchasing tickets for a user-determined destination (Jameson, 2009). In both these instances, the goals are again user driven and the interaction is open ended from the system's perspective. Specifically, different users choose (implicitly or explicitly) different email sorting and scheduling strategies in the automation system, and in the dialogue systems different users are directed to different goals (i.e. destination). More interestingly in spoken dialogue systems user's self-direct towards different

transaction types—careful and thorough sessions for novice users, and quick and efficient sessions for experienced users (Jameson, 2009). Thus, in both of these system types, a big role of the user modeling system is to identify a user's goal among a plethora of possibilities and adapt the system to suit it. Both systems are open-ended in that user's decide where the end point of the interaction is.

Each of the examples above illustrates a distinct application of user modeling systems and their associated adaptable systems. There exist other uses of user modeling that do not fit into a specific application camp but instead apply to many types of applications. Two examples are adapting the interface and changing the system presentation based on the user model. Examples of these techniques include slowing the registration rate of keystrokes for the elderly, or placing the most used tools for a productivity system in plain sight (Jameson, 2009). These two concepts extend throughout every problem domain of user modeling as they are basic examples of techniques by which user models can be put to use and we too will try to leverage their ideas in future work.

In contrast to the domains listed above, users of our system have **singular, fixed goal,** which they cannot avoid – the task of survey completion. We cannot offer alternative objectives, or suggest different sets of questions they find more interesting; instead we must create systems that guide the user through a fixed set of not always pleasant tasks, and take actions that subtly encourage them towards the end goal; which one could argue, is of more utility to the survey administer than the survey taker themselves.

Recall that our current models aim to guide by detecting and averting rare-user actions (i.e. breakoff in the Gallup panel, and memory failure in ATUS) hostile to the singular, fixed goal. Another distinction, between the system presented in this thesis and more traditional systems, is that the behavior we are trying to detect is not normal system behavior, but it does resemble it. User models often attempt to guide users towards beneficial behaviors, and thus if they see users engaging in costly behavior they can attempt to address the users actions by redirecting them. In contrast, however, one of the mistakes we address is final and irreversible. For example, for the Gallup survey, once you're gone (from a survey) you can't go back (i.e., breakoff error). The memory failure error of ATUS, on the other hand, may be equally addressable both once its observed, and prior to its occurrence. However, identifying failures early would enable the survey mechanism to engage in pre-emptive memory queuing, or preemptive interface adaptation that allows the survey to proceed more carefully. It would also provide domain experts, such as psychologists with more options in addressing memory failure as it could be preemptively attacked. Consequently, our models actually need to infer that users are taking poor actions in the near future instead of detecting such things and redirecting users.

The rare nature of the errors also leads to a large class imbalance problem which adds difficulty to the task of machine learning. We will discuss this more in Section 2.5, but for now note that the class imbalance makes it difficult for machine learning techniques to accurately engage the minority class. Consequently, we employ resampling techniques and a two-tiered classification method to address the issue.

Speaking of rare actions, it is also important to note another field of machine learning applications—anomaly detection. Anomaly detection utilizes machine learning or data mining to detect states of a system that are rare such as security violations (Salem and Stolfo, 2011) or credit card fraud (Chandola et al, 2009). It is defined as finding "patterns in data that do not conform to a well-defined notion of normal behavior" (Chandola et al, 2009). Given that the errors we detect are expected, they cannot be said to be anomalous but rather rare and costly user errors that are valuable to address. Furthermore, the goal of anomaly detection is often to warn the system about a threat or danger and *not* to assist the user. For these reasons, we do not think anomaly detection is an appropriate way to describe the rare-action problem, nor is it an appropriate framework as the user's committing these actions are to be assisted and do not display markedly different, i.e. anomalous, behavior.

Thus, one can see that our task is quite different from traditional user modeling problems. We can however, still make use of the user modeling paradigm, as the structures of such models fit very nicely on the frame of our problem—predicting user intentions and assisting them in accomplishing a goal.

2.2 User modeling techniques

User models base their user descriptions on observations of users. These observations arrive in two forms—explicit feedback and settings provided directly by the users themselves, or implicit conclusions about users derived from their actions, traits, or past behaviors.

Explicit feedback can be used to base a model on preferences, expressed interests, or similar attributes. For example, explicit ratings are used in systems such as Netflix and

Amazon in order to produce future recommendations to a user. Online newspapers suggest articles to users similarly, by basing recommendations off explicit user flags.

To draw implicit conclusions and adapt models, systems often leverage machine learning and data mining techniques. Indeed, this is a very popular approach which has been employed in a variety of settings. For example, Englebrech et al (2009) used hidden Markov models to model users based on satisfaction in spoken dialogue systems, Virvou et al (2012) used k-means clustering to model students in tutoring systems, and Pennacchiotti and Popescu used machine learning techniques to classify twitter users (2011). Machine learning techniques in particular offer the advantage of taking diverse data and developing a classification based on the observations. Therefore, if the designer has access to labeled data, supervised learning presents an attractive opportunity towards user categorization.

Another technique is to use basic statistical relationships (Jameson, 2009). For example, in recommendation systems, adaptation can be based on statistics such as x% of users who purchase a product Y also purchasing product Z. In this way, users can be offered products that have some association with the product currently being examined.

Another interesting way of viewing how models are derived is the fact that some techniques rely only on the current, individual users, and others make use of similar or related users. Collaborative filtering, for instance, creates suggestions or personalization for users based on other people who are in some way associated or deemed similar to the current individual. These types of techniques can be applied in a variety of settings, from information acquisition tools to recommendation systems. In contrast, other systems such as task automation, base their model solely on the user's preferences and actions.

Another dichotomy in user modeling is whether to model each individual, or to stereotype users into categories. The choice is contingent on the goals of the modeling and the purpose of the system. For a commercial application, it is obviously desirable to tailor presentation of products based on the customers' own records, whereas for systems that do not necessarily have repeat visitors or in which the user's identity is not as relevant (such as spoken dialogue systems) it is more desirable to identify a trait of the user that fits them into a certain category (such as the user being frustrated with the pace of the dialogue and thus needing assistance via shortcuts).

In our system, we have chosen to go the stereotype route for both the Gallup and ATUS surveys. In ATUS, this is an obvious choice as each user only takes the survey once. In Gallup, this choice is currently more desirable as we only have a record of 7 surveys (6 of which we're actually studying) and every user is not administered every survey. Due to the shallow nature of the data for any user, it is currently more desirable to stereotype rather than individualize. The way we stereotype users is to classify them as rare actors or non-rare actors—that is, individuals who will commit an error and those who won't, respectively.

As far as observation technique is concerned, all that is available to us currently are implicit feedback mechanisms by "observing" user actions via log-files generated during the surveys. As such, machine learning techniques have been adopted to fit users into the two categories. Machine learning offers the best option as we have found rare-acting users to be superficially similar to non-rare acting users, and thus we need the inference abilities of machine learning to help us sort out the differences between the two camps.

In the future, which will be discussed more in the final chapter of this thesis, we can put explicit observations to work in updating our models.

2.3 Why User Modeling?

In survey or interview systems, errors are abound. Software redesign can prevent errors beforehand in many systems (Curzon and Blandford, 2001), and survey or interview systems are no different. These errors include:

- ***Inconsistencies in user responses***. In ATUS, for example, users recall the events of their previous day in a time diary. As such it is possible for users to provide inconsistent information. Consider an activity sequence in a time diary like: watching TV at home, grocery shopping at Walmart. Here, we see the user has forgotten to include how they traveled from their home to the store. A basic logic engine that checks for sequential activities occurring in different locations can find these errors and bring them to the user's attention.
- ***Omitting information***. In ATUS each entry in the time-diary includes information such as what the user was doing, where they did it, and with whom they performed the activity. Omitting information in ATUS would mean omitting data in the time-diary such as where the activity occurred, with whom it occurred, or other required information. In Gallup (which is simply a series of questions), this would mean a user not answering part of a question. Missing information is easily checked, and can be brought to a user's attention.
- ***The absence of expected information***. In ATUS, for example, it is expected that any respondent likely slept, ate and groomed during the course of a day. If this

information is not spotted by the interview system it can be brought to the user's attention.

While redesign can prevent many user errors beforehand, (Curzon and Blandford, 2001) some errors are immitigable by redesign. These include errors that occur do to a conscious decision to avert a goal and errors that are due to a user failure that is not addressable by the machine. Two specific examples are users quitting surveys before completion in the Gallup panel (i.e., breakoff) and user's failing to remember activities in ATUS. There is no way for a software redesign to prevent a user's conscious decision to quit a survey or to prevent a user's memory from failing.

Predicting such errors before they occur, however, could allow an adaptive interface to take action that would possibly prevent the errors. Methods to prevent breakoff include presenting material more pleasing to the user (i.e., questions they deem more interesting as **extra** questions, we cannot **remove** unpleasant questions), offering clearer explanations regarding questions, or offering encouragement in order to induce in the user the idea that taking the survey is of worth. In regards to memory failures in ATUS, adaptive designs could take actions such as providing memory cues or proceeding in fashion more conducive to effective or efficient memory recall. Domain expertise provided by psychologists or other social scientists will be needed in designing effective mechanisms addressing the ATUS rare-action.

The implication is thus that we are building user models to address one problem in each system. While we can extend these models to address more issues such as comprehension of questions, one might question whether such a narrow window truly

constitutes a user modeling problem. Fischer points out that often user models in fact serve only one or two purposes – and are intended to assist a user in a narrow scope within the program. Therefore, we see that we can indeed classify this work into user modeling, even if it is not extended further in the near future.

Finally, note that in our work, we also mine statistics related to rare-actions (i.e. errors and goal damaging behavior) based on user behavior, user characteristics, and system characteristics in order to inform and support survey or interview redesign and hence aid users before they even engage with the systems. This is thus part of the larger user modeling problem. The findings made in this fashion also have implications regarding the design of the ML predictive mechanisms used to detect rare-actions, which we will discuss in the Results and Future Works Chapters of this thesis.

2.4 The Process of User Modeling

Frias-Martinez et al (2006) break the steps of user modeling into four phases:

- Data collection
- Data preprocessing
- Pattern recognition
- Validation and interpretation

This thesis engages with each of these phases, but deals mainly with the first three. Data collection involves gathering the data necessary for construction of a model. This step was accomplished by building a database representing Gallup surveys, user demographics, user responses and user interactions, and by parsing ATUS files communicating user actions, demographics, and response patterns. Note that we do not

build a representation a one size fits all representation of ATUS surveys (as we do in Gallup), as they are time-diary interviews in which the respondent catalogues their actions throughout a day. We instead build the sequential list of activities each user describes.

Data preprocessing involves getting the data into a state useful for pattern recognition purposes. In this thesis, we accomplished this by rebuilding user sessions and logging relevant semantic actions (as Frias-Martinez et al (2006) point out this is a common practice in user modeling) and by normalizing data using techniques explained and justified in Chapter 3.

With the data preprocessed, we engaged in pattern recognition via machine learning amplified with a secondary rule scheme, which was used to identify users destined to engage in rare actions. The role of the rule scheme is to override ML classification decisions, based on domain knowledge contained in two rule sets – one rule-set used to identify rare actors and hence overturn non-rare action classifications, and one rule-set used to identify non-rare actors and hence overturn rare-action classifications. ML techniques are often used in this phase as they are able to translate data into a structural form capable of classifying said data, and hence users, into distinct categories (Frias-Martinez et al, 2006). Furthermore, developing rules containing domain knowledge has been done in user modeling problems in the past (Frias-Martinez set al, 2006).

Although we can see that ML techniques are a rational decision for the pattern recognition phase, an open question is what classifier is best suited to our problem. As in any ML problem, several considerations must be applied:

- What is the purpose of the ML technique? In our case, we hope to derive an output, which indicates whether a user is a rare action user or a non-rare action user.
- Are slow training times acceptable? Given that models are likely to need to change over time and that new models will likely be needed for each survey or even for each survey page, it is questionable whether slow training times are acceptable. The way surveys are administered users often have access to the same survey during the same timeframe. In order to maximize the model's utility, it is desirable to offer it to as many users as possible. There are several options here. We could
    - Attempt to fit models to pages or parts of a survey that have already been created based on how similar a page is deemed to a previous page. In this way, we could offer a model from the start, and build a new model over time, as this one proves reliable or faulty. More interestingly, this would offer a chance to develop a maturation process for the rules in which rule sets are gradually refitted or further refined as a given model is applied to many surveys. This type of technique would avoid the training time question.
    - Not provide early users with any model, but instead offer them an alternative form or assistance or even an alternative model not dependent on machine learning. This could lead to comparative studies in which we can attempt to conclude whether one of the two approaches displays superior performance. It is quite possible after all; that a sound adaptable

or adaptive design might hinge on offering the user more opportunities to interpret the survey or alter its presentation than currently available, and that prediction is unnecessary (Conrad et al, 2007). Work done toward prediction should not omit such designs from our consideration.

Note that slower training times on large datasets could render the model unusable to an unacceptable percentage of the users. In this case we would definitely want to avoid such methods. Thus we consider this question still open, and have not hesitated to explore slow learning classifiers and resampling techniques in testing, such as artificial neural networks and ensemble systems of resampling.

- Is there a desire to interpret the output of the model? This could be of interest to us as we are trying to build models that explain why users engage in rare actions to some capacity. Thus, it may be more valuable to employ techniques with readable output. As such, we have explored decision trees and of course, rules.

The final phase involves interpreting the results of the model and validating the model. The most we can do this now is o to test whether our predictive capacities are adequate. Thus, we have done so by determining at what level we are able to predict rare action. To make this difficult, the very fact that the actions we are attempting to predict are rare leads to an extreme class imbalance problem in our dataset, making it very difficult to develop classifiers that identify the minority class at levels traditionally deemed acceptable.

2.5 The class imbalance problem

(A couple of terms we use throughout this section are sensitivity to class balance, and insensitivity to class balance. By sensitivity to balance we mean that toggling the balance will alter the value of a statistic even though the classifier itself does not change (He and Garcia, 2009). A statistic is called insensitive, on the other hand, if its value only shifts when the underlying classifier model is modified (He and Garcia, 2009).)

In order to address the class imbalance problem, imbalanced data techniques were applied to the data that the classifier received. Imbalanced techniques necessitate from the fact that machine learning classifiers generally attempt to maximize accuracy – a statistic ignorant of the relative costs of misclassifying different types of data (He and Garcia, 2009). As such, the performance of a minority class suffers if the data is imbalanced and the minority class is not obviously distinct from the majority class.

To address this issue, resampling techniques and cost techniques have been developed by researchers. Since we have no non-arbitrary quantitative measure of cost, we employ resampling techniques to address the imbalance problem. We describe the specific techniques employed in Chapter 3.

Generally though, resampling techniques work by altering the class balance present in the training data. This can be done by oversampling the minority class, undersampling the majority class, or some combination of both. Newer techniques, such as SMOTE, create new, synthetic examples, in order to balance the training set (Chawla et al, 2002). Despite any class balancing in the training set, the sets on which classifiers are tested should be independent and reflect the true class balance in order to get an idea of true classifier performance.

The way we treat our data is such that we have a two class machine learning problem. Each data point is labeled as either being a case of rare action or not a case of rare action. Thus, our minority class is the rare action data and our majority class is the non-rare action data. Currently, our class balances range from the rare data occupying 4.5% of the total dataset in ATUS, and the rare data occupying 2.3% of the dataset and less in Gallup. The reason the balance lessons within Gallup is because as a survey proceeds fewer breakoff users remain in Gallup, and hence the imbalance increases over time.

Another question is how to quantify classifier performance in the presence of large class imbalances. Again, the issue is that accuracy numbers do not reflect how well a classifier performs on each class. Instead, it gives an overall measure of correctness which is extremely sensitive to class balance (He and Garcia, 2009); that is if a classifier performs extremely well on one class and very poorly on another class, the state of the balance between these two classes has a great effect on the measure of accuracy.

Due to this, it is desirable to measure performance on imbalanced data with measures that take into account the balance. One such measure is the geometric mean (g-mean) of the true positive rate and true negative rate (hence the square root of the accuracy on only the positive class multiplied by the accuracy on only the negative class). This measure is not sensitive to class balance – that is if the true positive rate and true negative rate are constant, a shift in class balance will not effect the g-mean. Furthermore, the g-mean will only deliver solid values if a classifier has a true positive and true negative rate both performing well.

Other measures we are interested in are recall and precision defined in terms of what we call the positive class – cases of rare action. Given the extreme imbalance of our datasets, it is very difficult to achieve high precisions – there are simply so many more non-rare cases than rare cases that even a misidentification of just 5% of negative cases would mean for most Gallup panel datasets that we have 1000 false positives, a number substantially greater than all of the rare cases in most datasets, which usually have around 300 such cases. Thus, a precision of 30% would represent extremely good performance, as we would have 100% recall and a 95% accuracy on the negative class.

This situation is due to the fact that precision too is sensitive to class balance as the precision statistic is dependent on "both sides" of the confusion matrix; thus if the class imbalance is extreme, the amount of false positives become much more numerous, although **not necessarily** more likely when we view the false positive statistic in terms of the false positive rate (He and Garcia, 2009). Thus, in our experiments we tend to downplay the precision statistic (although we no doubt want to improve it using techniques we apply and in future work) in favor of the g-mean and recall statistic, neither of which are sensitive to class balance.

Recall is not sensitive to class balance as it represents the "completeness" with which one classifies the positive cases, and its value is therefore dependent only on how completely a classifier categorizes positive cases (He and Garcia, 2009). In other words, it is only dependent on well the classifier performs on a single class, and thus any performance of the negative class has nothing to do with the recall statistic.

We obviously still need some way to measure how well we perform on the negative class, however, and that is why we chose the g-mean; as that statistic reflects the performance on both classes, insensitive to the class balance.

Another measure we employ is the Matthew's correlation coefficient (MCC). The MCC is essentially the Pearson correlation statistic applied to assessing the dependence between the output of a ML classifier and the actual value of the data (Baldi et al, 2000). A value of +1 indicates that the ML classifier completely agrees with the actual values, a value or -1 indicates the ML classifier completely disagrees with the actual values, and a value of 0 indicates the ML classifier is making completely random predictions (Baldi et al, 2000). We can thus use the MCC to tell if our classifiers are differ from random guessing, and can test this condition statistically via the MCC's relationship to the Chi-squared statistic which we will discuss in Chapter 5.

2.6 Survey error overlap and rule implications

A large-part of this thesis involves the exploitation of domain knowledge in the creation of rule-sets used to boost classifier performance. Relevant to this endeavor is the fact that issues users encounter in surveys can be catalogued based on their **a) sources**, and **b) causes**. Significantly, there is large degree of overlap between the causes of many errors, even regardless of their source. In this section we will discuss sources and causes of user-error in surveys and synthesize a relationship between different types of error and a list of common sources. We will then illustrate the implications these observations have for rule-set generation.

All non-response errors are thought to be caused by problems of motivation in respondents, opportunity of respondents to answer the questions, and the abilities of

respondents to answer the questions (Bosnjak and Tuten, 2001). The source of these issues can stem from the technology used to collect the data, actions of the interviewer (if present), or traits of the respondent themselves (namely in the cases of motivation and comprehension) (Bosnjak and Tuten, 2001; Krosnick, 1991; Nichols et al, 2012).

Measurement error is more complex to pinpoint as its manifestations are well-defined but difficult to say with certainty whether the manifestation is a coincidence, or an actual instance of a measurement error occurring. For instance, one measurement error manifestation is **straightlining**, which is defined as users bubbling in the same answer to a series of multiple choice questions to save time or effort (Krosnick, 1991). While such events imply that a measurement error might be occurring, there is no way for software to know in real time. We do have an opportunity when observing such manifestations, however, as measurement errors share many of the same theoretical causes and sources as non-response error.

The theoretical causes of measurement error are motivation, deliberate distortion, and poor user comprehension (Couper, 2000). Theorized sources lay with the user, poor question wording, poor question design, and technical difficulties (Couper, 2000). Satisficing – defined as user effort degrading during a survey or interview – is another theorized cause of error and is closely related to the motivational and comprehension issues (Krosnick, 1991).

Note, the significant overlap between causes and sources of non-response and measurement errors:

- Both error types can be caused by cognitive difficulties encountered by the user (i.e., comprehension, and cognitive effort declining)

- Both error types can be caused by motivational problems.
- Both error types can be sourced from technical problems.
- Both error types can be sourced from the respondents themselves.

From a user modeling perspective, these indicators of each cause and effect are valuable targets for both data collection and rule-generation as they contain information regarding the likelihood a user has in taking a rare-action. Note also, that since causes and sources of error types overlap the act of simply observing potential (though not certain) measurement errors like straightlining have the potential to inform a modeling system aimed towards other errors, and thus the rare actions we seek to detect.

Furthermore, depending on the specific issues the user is encountering—for example, cognitive issues vs. motivation, survey methodologists expect different, and even **contrary**, domain heuristics to hold. For example, it is also theorized that certain conditions "foster satisficing" (Krosnick, 1991) consequently effecting causes of measurement and non-response errors. Such conditions include the difficulty of the task (which can be related back to survey layout and question topic, complexity, and word length) and the length of the survey which can be linked back to motivational problems. These issues lead to contrary expectations from survey methodologists. In the case of motivation, younger users are expected be more effected (Peytchev, 2009), and in the case of task difficulty, older respondents are expected to be effected as cognitive ability is expected to decline with age (Knäuper et al, 1997). Therefore, in certain contexts domain knowledge flags older users as likely offenders, but flags younger users in other contexts.

This implies that our rule-based system must be sensitive to the current context of the survey in order to properly address user-needs. We then face two choices:

1. Create a system capable of understanding the ramifications the current system context is likely to have from users, and base the rules on this.
2. Address the context problem by allowing the rules to seek their precise meanings in an exploratory fashion; thus leaving the system open to exploiting any particular context without needing an explicit representation.

We have chosen the latter course for the time being, and in the following chapters the design and implementation will be discussed further.

2.7 Summary

In summary, this thesis builds user models following a pattern well defined for the task: data collection, data preprocessing, pattern recognition, and interpretation/assessment. We aim our modeling towards recognizing rare forms of user action, which is distinct from the goals of most user models, and build our models using machine learning techniques, domain heuristic based rules, and resampling techniques proven to address imbalanced datasets. The user models also are designed for an environment different than most applications in that goals are not self-directed, user navigation of content is limited, and the content set itself is restricted. Currently, we can only assess our pattern recognition and not our model in practice, and we do so using statistics that other researchers have illustrated to be insensitive to imbalanced datasets.

Chapter 3

Methodology

The twin aims of this thesis are to develop user models capable of predicting rare user actions, and extracting features of the system that are associated with these rare actions. The methodology addresses these twin aims by: 1) applying machine learning classifiers employing resampling techniques and equipped with a secondary, refining, rule technique which is used to identify rare-action users in hostile environment; and 2) performing simple data mining on survey features in order to extract attributes associated with error. In this chapter we describe the details of this methodology.

3.1 Modeling rare user actions

In order to model user behavior pertaining to rare actions, machine learning algorithms were applied to user data as a way of predicting future actions based on a feature set composed of past actions, user demographics, and survey features. These algorithms employed the use of imbalanced learning techniques, due to our data having a large class

imbalance. These classifiers were then augmented with a rule-technique in order to improve performance.

The rule-based technique is used to improve performance because, due to the severe imbalanced learning problem, there is just not enough information or knowledge contained in the dataset for it to be learned, and thus an effective classifier to be obtained. As a result, we look to injecting domain expertise as rules or heuristics into the system.

In general, the rules are based on domain knowledge and human expertise. Specifically, they are based on re-examining attributes known to be theoretically associated with the specific rare actions we are trying to prevent with our system. The developed rule system overturns classifications if the rule system claims there is enough support for such a decision. Using this technique, we are able to boost the g-mean measure of classifier performance, consequently improving the rate at which we distinguish rare-action users from non-rare action users.

To generate the rules, we have two methods. Dealing with nominal attributes, we use domain knowledge or heuristics to devise the rules directly, as it is rather straightforward to map nominal attributes this domain knowledge. However, when dealing with numeric attributes, as alluded to earlier in Chapter 2 (Section 2.6) whether an attribute-value should be considered associated with rare-action is partly dependent on context. Thus, in order to condition rules on numeric attributes, actual boundaries, or regions of values, for each numeric attribute have to be identified that separate rare-action users from ordinary users. Note that these boundaries can shift as the environment of the survey changes.

To handle numeric attributes, a rule-generation process creates rules by exploring which attribute-value regions are most associated with the presence or absence of rare-actions. These regions can then be projected into a rule described by a comparison operator and boundary-value which segregates the region of interest from the other data. In our domain context, differences in numeric attribute-values often convey, according to literature, a difference in an underlying user-emotive or cognitive state such as user-motivation or attention. These user-mindset differences impact the presence of not only general survey errors, but also the specific rare actions we hunt for. Furthermore, different survey contexts vary in the amount of focus and motivation required by users to adequately complete the tasks; and thus rather than pre-ordaining the directional relationships rules should seek by examining each survey context in detail, we instead allow a rule exploration process to search each different potential region. This allows us to implicitly take context into consideration.

We also consider exploring attribute-value regions of nominal attributes in a categorical-static fashion. Unlike the numerical attribute-vale pairs, nominal attribute-value pairs can be placed into groups conveying meaningful, abstract concepts ahead of the search process. For example, certain operating systems are associated with mobile devices. Thus by ordering our rule search to examine specific combinations of operating system values, such that the combinations indicate the user was employing a mobile operating system, we can explore rules containing reasonable, meaningful knowledge. In this example, that would mean exploring whether mobile device users are more likely to commit rare-actions.

However, leaving these categorical searches wide-open risks identifying noisy, inconsistent rules that are neither based on, nor exploitative of domain knowledge. Thus, we prescribe the combinations of attribute-value pairs to search for these nominal attributes, narrowing the "search space", and conducting the exploration in a "guarded" manner. In other words, the nominal attribute rule search is guided directly towards attribute-value combinations derived from domain knowledge.

More details regarding rule generation are presented in Section 3.1.9.

3.1.1 Features

The feature set used in our user models is composed of past actions, user demographics, and survey features. Below we justify the use of each of each set of features.

1. Past Actions – Distinguish normal user behavior from behavior associated with rare actions.
2. Demographics – Identify groups of users inclined towards rare actions.
3. Software Features – Identify context features capable of inducing rare actions in users.

From a survey methodology perspective, each member of the feature set can be justified theoretically from the literature. The list below lays out these justifications:

1. Past Actions – user actions can be indicative of satisficing, difficulty understanding questions, speeding, and other indicators of sub-optimal survey behavior (Krosnick, 1991; Galesic and Bosnjak, 2009).
2. Demographics – certain user groups are associated with survey error, one example is individuals with less time spent in formal education institutions (Peytchev, 2009; Johnson et al, 2010; Guittierez et al, 2011).

3. Survey Features – grid questions (a special type of multiple choice question composed of a common stem with child questions altering only one part of the stem question; for example, rate each these Nebraskan restaurants on a scale from 1 to 10: Runza, Amigos, etc.), question word count, and number of questions on a page have all been associated with errors (Fricker et al, 2005; Galesic and Bosnjak, 2009).

In ATUS we explicitly represent survey features by measuring the current state of time diaries, while in the Gallup panel our models implicitly incorporate survey features via normalization techniques and because we build a distinct dataset for each distinct context (i.e. survey page) in Gallup.

3.1.2 Feature Context

A context, or frame of reference must also be established for certain sets of features, (namely past actions, and software features in this thesis) as the distribution of values associated with features can change in appearance depending on the context from which the values are extracted. With this in mind, we argue that any software interaction can be viewed from a set of scopes ranging from very low level user actions to the entire history of a users interactions with a piece of software.

This thesis has applied this framework to the rare action user modeling problem in surveys and we argue that it could be applied to other endeavors as well. Below, we define this framework and go on to describe its application to the practical problem under study in this thesis.

3.1.3 A framework for user modeling

We argue that user behavior can be tracked from various scopes within a software application. Namely:

1. Action Level – based on the minute, small actions a user performs; such as keystrokes, or mouse movements.
2. Session Level – based on more abstracted actions that have an impact on the outcome of a complete session of software use. For example, a user purchasing goods from an online retailer represents an abstract action of significant consequence to the user and vendor during a user's given session.
3. Application Level – actions drawn from the entire history of a user's interaction with a software system. Examples might include, the frequency at which a user logs into a system, or the amount of times they have performed certain session level actions in the past.

Using this framework, we can fit breakoff in the Gallup panel into the Session Level scope as the action of quitting a survey renders the users entire interaction worthless from a data gathering perspective. Memory gaps in the ATUS fit into the session level category as well as they represent a failure to complete a certain section of the survey, and thus represent a more abstracted action than say, responding to a direct interviewer question or asking an interviewer a clarifying question.

Given this session-level context (or another context, potentially), the features defined in section 3.1.1 can be normalized. By normalization, we mean refactoring the data such that values are defined not as absolute values or raw counts, but absolute or raw values divided by some normalization factor or measured within a certain window. This factor can be based on values other users tend to have, or values this user usually has, or

any other such measure so long as it can be applied to all users. In this way, we can determine if the user is straying from a typical user's course of action, or if they are straying from their own typical course of action, or how their behavior varies in regards to some other context. For example, a third way to normalize is to treat a page of Gallup as the frame of reference, and thus for each user calculate actions per page, rather than simply raw action counts. Normalization in this manner can provide insight into a user's typical course of actions and therefore be less sensitive to noisy shifts in user behavior. Using these normalization techniques, we can study whether any particular tactic leads to improved classification performance.

Given this ability, many different forms of any dataset can be generated. For our practical problem of predicting survey error, this normalization was performed for Gallup in four different ways, and was performed in ATUS in two different ways. In ATUS we also investigated datasets using no normalizations.

We articulate these normalizations in the subsections below.

3.1.4 Normalization in Gallup

As described in Chapter 1, Gallup surveys are administered via the Internet and questions appear on a page by page basis. Thus from a session level perspective, we can target user behaviors based on their actions on a certain page. Because users must all deal with the same content on a given page, measuring user behavior in Gallup based on the current page implicitly incorporates the context of the survey. Even for attributes that include information about past user behaviors implicitly incorporate context for the same reason – all users are forced to have the same context history as all users must engage the same sequence of pages.

User behavior can be normalized by page in many ways. One can average user behavior based on the number of actions per page. On can also normalize a users behavior by dividing it by the average user behavior on a given page. Another way is to take the user behavior on the current page and divide it by the user's own average behavior on all past pages of the survey. Each of these methods were applied to the Gallup data and we justify their use in the list below:

1. Actions per page – each page in the Gallup panel consists of a different set of questions, and thus a different environment. One way to normalize the data is to record how many actions a user engages per page. We can then center the actions per page into a series of distinct datasets, each one representing the behavioral values each user has generated up to a given page in the survey. The derived data is then a series of datasets each one showing what a user's data looked like when they engaged with a unique page.
2. Actions per page (but based on page order) – this strategy is identical to the first option, but builds datasets centered on users currently being at the same number of pages into the survey. This approach was used because it puts users all into the same context of number of past pages seen and thus has implications both for how tired of the survey they may be, and how many opportunities they've had to accrue various actions.
3. Actions versus average – normalizing a user's actions over the average user action allows us to develop measures that indicate how non-normative a user's behavior is. Since this thesis is concerned with rare user actions, such a view of the data could potentially be beneficial. This technique also develops a

series datasets centered on each unique page in the survey, in order to implicitly take context into consideration.

4. Actions versus a user's own past actions – rare user actions could be due to the degradation of user's experience over time, or due to some other long-term shift in their experience. By normalizing user actions over their past actions we can develop measures that given an insight into this phenomena. Furthermore, individual people likely have vastly different tendencies of software use dependent on features wholly unknowable to the current software system – such as personality, amount of last night's sleep, and all sorts of other traits. Thus, one might argue a better way to compare users to each other is to develop measures that are based on normalizing a user's current behavior to their average behavior. Such a measure can help us overcome differences in innate tendencies. This technique also develops a series datasets centered on each unique page in the survey, in order to implicitly take context into consideration.

Note that the implication of the normalization techniques described above is that the Gallup model consists of series of classifiers, each one assigned to predict rare-action on a current page. It should also be noted that each Gallup classifier attempts to identify all future rare-actors, not just the rare-actors who commit the action on the page they are centered on.

3.1.5 Normalization in ATUS

ATUS is a very different beast from Gallup in terms of normalization techniques that can be applied. This is due to the fact that there are no set questions in ATUS but instead the

user describes a sequence of behaviors they engaged in in the prior day. Due to this, we can normalize the count of certain action values to the total number of activities, use no normalization at all and instead reflect the raw measurements, or normalize the average amount of certain actions in all users. These normalization techniques were applied to the ATUS data.

3.1.6 Machine learning for class imbalance and resampling

With datasets constructed for session level behavior and based on a variety of normalization schemes, our methodology then applies machine learning (ML) techniques to these datasets. The class imbalance problem inherent to the rare user actions problem makes successful application of such ML-based classifiers very difficult because standard ML-based classifiers attempt to maximize accuracy or ignore costs of misclassifying various classes (He and Garcia, 2009). Sensitivity to the accuracy statistic and ignoring misclassification costs causes classifiers to be constructed that perform very well on the majority class, but very poorly on the minority class. This is an artifact of the imbalance itself, as classifiers are able to maximize accuracy by learning to perform very well on only the majority class. He and Garcia point that it is common for such classifiers to yield "close to 100% accuracy on the majority class while delivering "accuracies of 0-10%" on the minority class (He and Garcia, 2009).

This difficulty is amplified by the fact that such actions in our problem context are related to either a user's cognitive state and decision making. It is very difficult to detect moods, frustration, or other factors that may lead to quitting or memory failure using machine learning classifiers. For example, Kapoor et al (2007) were able to detect frustration in a tutoring system at a rate of 79% (with 58% of the data being positive for

frustration) using a combination of physiological and visual measures. We argue that detecting the practical issues of survey breakoff or memory failure is similar to such studies as it is a matter of detecting some state of mind of the user; and that the class imbalance problem increases this already difficult task.

In our case, the class imbalance problem constitutes imbalances at 2.3% and below for the Gallup panel, and 4.5% for ATUS. Note that for Gallup, the class imbalance problem grows as the survey continues. This happens because as breakoff users drop out of the survey, fewer remain compared to the population of normal behavior users, driving the class imbalance problem down to percentages under 1%. Such imbalances represent a significant obstacle to machine learning algorithms, and, as a consequence, successful user modeling capable of predicting rare actions. Since an important part of these models is to accurately predict rare actions, techniques additional must be applied to machine learning classifiers capable of addressing the class imbalance and cognitive complexity problems.

Given the data available to us, the class imbalance problem can be addressed whereas gaining insight into a user's cognitive state and decision-making state is not possible. We addressed class imbalance by applying well-known techniques. These techniques fall into two camps: resampling techniques and ensemble-based classifiers derived from resampling. For resampling, we applied undersampling and oversampling. For ensemble-based classifiers we applied the Balanced Cascade algorithm and the Easy Ensemble algorithm described in Liu et al, 2009**.**

Resampling techniques randomly resample the dataset in order to artificially alter the class balance. For undersampling, a subset of the majority class is subsampled and

amended to most or all of the minority class cases and this new dataset is then used in training. Oversampling works by resampling the minority class itself, so that multiple instances of a given minority class data point will appear more than once in the training set. This new set is then amended to all or a subset of the majority class cases.

The ensemble-based techniques are also based on resampling. Both the Easy Ensemble algorithm and Balanced Cascade algorithm work by creating an ensemble of Adaboost classifiers each of which is trained with an undersampled subset of the original dataset (Liu, et al 2009). Each undersampled subset of the majority class is resampled only from those cases that some member of the ensemble does not currently correctly classify (Liu et al, 2009). The only difference between the two algorithms is that each Adaboost classifier's threshold is adjusted until it achieves a certain false positive rate on the undersampled set it is learning on in the Balanced Cascade algorithm (Liu et al, 2009).

These techniques were chosen because they have proven successful in many class imbalance problems in the past. Notably, we excluded the use of SMOTE (Chawla et al, 2002) as initial testing of SMOTE did not yield promising results. For example, applying SMOTE to a Gallup dataset in which the values were normalized against a user's average behavior yielded an accuracy of only 7.3% on cases of rare action. In gross terms this meant the system correctly identified only 9 / 232 rare action cases.

3.1.7 Rule-based filtering

Intelligent systems often leverage domain expertise and knowledge to improve classification performance or data mining (Sinha and Zhao 2008; Pohle, 2003; Bonissone et al, 2006). Doing so enables injection of human knowledge about complex systems into

machine learning and data mining techniques and has been shown to boost performance. We too have adopted this approach in order to improve the performance of our system, as classifier performance stagnated when utilizing classifiers and imbalance techniques only.

The form of domain expertise we have chosen is based on a system of rules that vote on whether a given user is likely to engage in rare actions (or not). The formation of this rule system was motivated by the stagnant classifier performance mentioned. Examples of similar systems include a set of rules developed to improve classification of bank loans (Sinha and Zhao, 2008). Using the rule technique, we are able to improve recall, the false positive rate, and the G-mean.

Stagnant classifier performance was observed from the fact that while recall was achieving levels at 65% or above; the false positive rate seemed to be stagnant at around 33% (thus the G-mean was stuck around 66% as well). Due to this, the rule-based filtering technique was applied to the output of the classifiers based on a set of rules created from domain expertise about our datasets. This rule technique identifies false positives and false negatives, and then changes their label to the correct value at a rate better than randomly reassigning data labels. The rule set yields an improved g-mean.

3.1.8 Rule set structure

We empower a given rule set to either turn rare action classifications to non-rare action classifications, or vice versa. Each rule set is responsible rendering a verdict on exactly one of these conversions. N rules compose a rule set. Each rule in a set casts a vote on whether a given datum should be reclassified by comparing a specific attribute value to a

baseline via a comparison operator. If the votes exceed a threshold, t, associated with the rule set, the datum is reclassified to the opposite of the current classification.

The implemented comparison operators are whether a value is:

- Less than or equal to a baseline
- Greater than or equal to a baseline
- Less than or equal to a baseline or greater than or equal to a second baseline
- Equal to a baseline
- Not equal to a baseline
- A member of a set of baselines
- Not a member of a set of baselines

3.1.9 How are the rules formed?

Rules are formed by first selecting attributes known to be associated with the rare action being predicted. Following this, the ranges of possible values for the attribute are explored as candidate rules. We leave the search open for numeric attributes due to the possibilities of different relationships between the attribute-value and the current survey context, but restrict the search for nominal attributes as these relationships are static. Each rule is based on domain knowledge found in the literature: either knowledge that a numeric measure is tied to an underlying source or rare action, or knowledge that a nominal category or a value is hypothesized to be linked to rare action. For example, it is known that age effects the likelihood to breakoff from surveys—hypothetically due to motivational issues—particularly that younger respondents are more prone to breakoff (Peytchev, 2009). Thus a rule whose utility we can investigate is to state that "if a respondent is younger than X years of age, then vote that they are a breakoff user."

However, age itself is also thought to be associated with cognitive ability and stamina. Hence, in other contexts (i.e., pages with difficult sets of questions or tasks) a more realistic rule would be "if a respondent is older than Y years of age, then vote that the user will perform a rare-action". Therefore, rather than prescribing the current context, we choose to leave the search open.

The precise cutoffs for values can be explored by moving this boundary-value (X or Y, in these examples) up or down, but they take their root from a common starting point; i.e., younger respondents are associated with motivational problems, and older respondents lack motivational problems (usually) but may experience difficulty with some tasks. Each approach leans on a theoretical association found in the survey literature between certain attributes and the rare action and thus can start our exploration of more precise values from there.

The rule sets can be further tuned by examining how each rule performs in isolation. If a given rule does not turn classifications at a rate better than random guessing (equivalent to the class imbalance) then the rule is discarded. Alternatively (and often equivalently), if the rule does not improve the current classifier's g-mean statistical measure, then the rule can be discarded. In this way, the rule set can be refined over time.

In summary, rule sets are formed in several phases:

1. The feature set is reviewed, and features known to be theoretically associated with the rare action are marked.
2. For each marked feature, a range of values are explored in a search for a specific range associated in the data with the presence of rare action, or the absence of the rare action. For example, older age is theoretically linked to a

lesser likelihood of breakoff. Consequently, we explore age-range based rules for specific rules that will improve classification, and hence user-model, performance.

3. Two sets of rules are generated. One set contains rules that state data with said attribute values are not likely to be rare action cases, and a contrasting set contains rules that state data with said attributes values are likely to be rare action cases.
4. The rule sets are tested over the range of all possible thresholds. Rule sets that convert classifications at a rate worse than randomly selecting data to re-label are adjusted to a stricter set of values. Once a rule cannot be readjusted further, it is discarded.

In the end, two rule sets exist. One rule set uses n_1 rules to convert classifications from rare actions to non-rare actions - we call this the True Negative Rule Set because its job is to improve the true negative rate. The second set uses n_2 rules to convert classifications from non-rare actions to rare actions – we call this the True Positive Rule Set, as its primary job is to improve the true positive rate. In each rule set, we can vary the threshold that must be exceeded for a reclassification to occur. The threshold can range from 0 (causing the rule set to always reclassify) to n+1 (causing the rule set to never reclassify data). Doing this, we can explore during testing to find the best threshold for each rule set.

Note that changing the threshold is equivalent to stating how many clauses conjunctive normal form of a subset of the rules must have and be satisfied before a

reclassification occurs. Thus, by changing the threshold we explore what number of rules must be successfully conducted for a reclassification to proceed.

We explore the best combination of rules further by varying whether the application of the True Positive Rule Set should be dependent on whether the True Negative Rule Set deems the datum to not be an instance of normal behavior. This version of the True Negative Rule Set is equipped with a threshold independent of the True Positive Rule Set, and the version of the True Negative Rule Set used to switch rare action cases to non-rare action cases. This was done because rare actions are very rare compared to normal behavior, and thus it is desired to explore a more stringent rule combination for reclassifying data from negative to positive.

In the end the application of the rules and classifiers mimics the structure depicted in Figure 3.1.

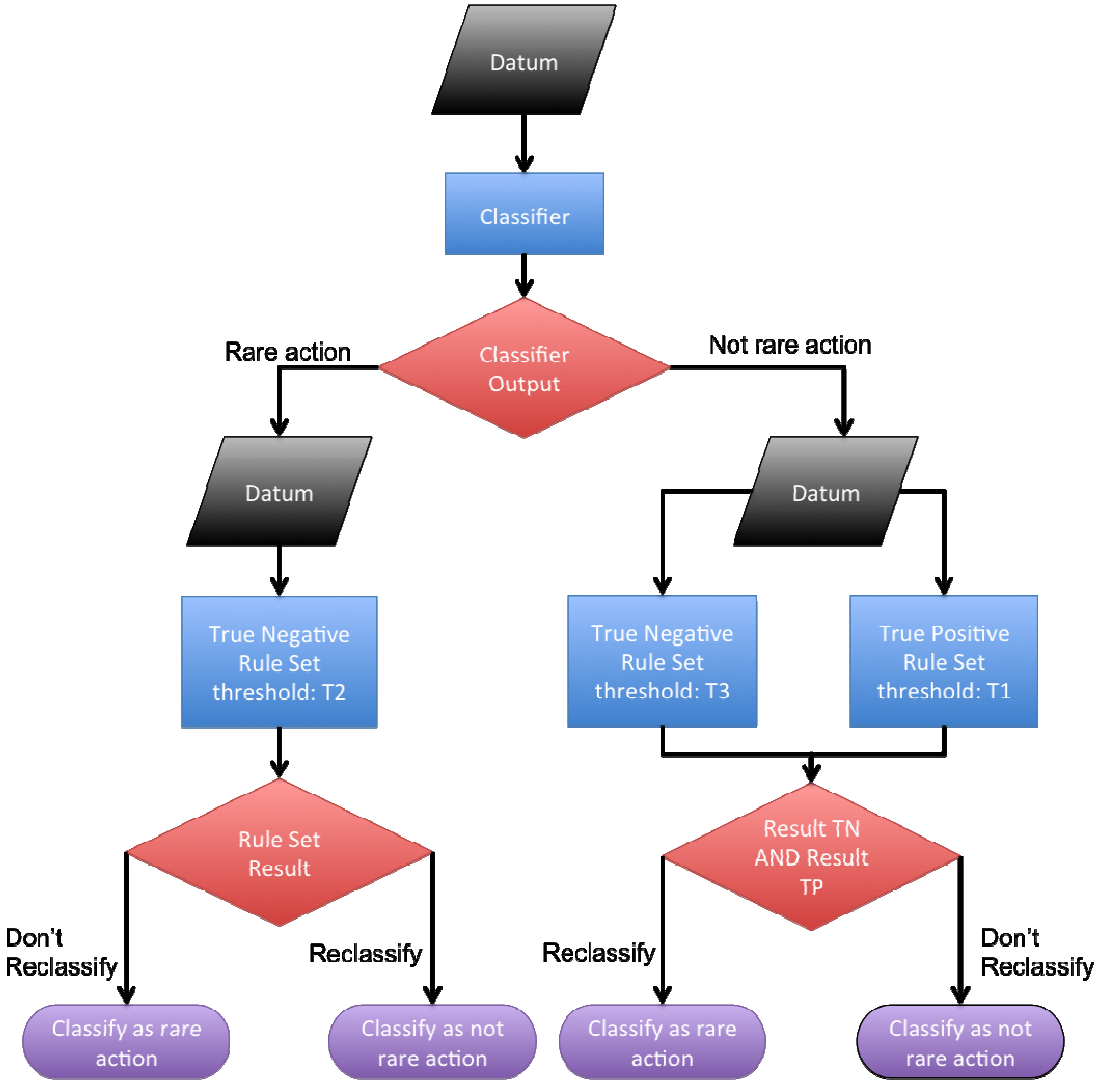

Figure 3.1 – Use of rule sets to filter classifier output. Each rule set makes use of an independent threshold in order to render its reclassification decision.

Using this system we are able to generate combinations of rules and classifiers that improve classification performance.

3.1.10 Overview of rare action detection

In summary, our system detects rare user actions by engaging in a process of:

1. Collecting relevant data about user behavior, survey features, and user demographics.
2. Normalizing said data.

3. Classifying data using data balancing and machine learning techniques.
4. Refining the ML classifications using a rule-based system.

The structure of our system can thus be viewed in Figure 3.2.

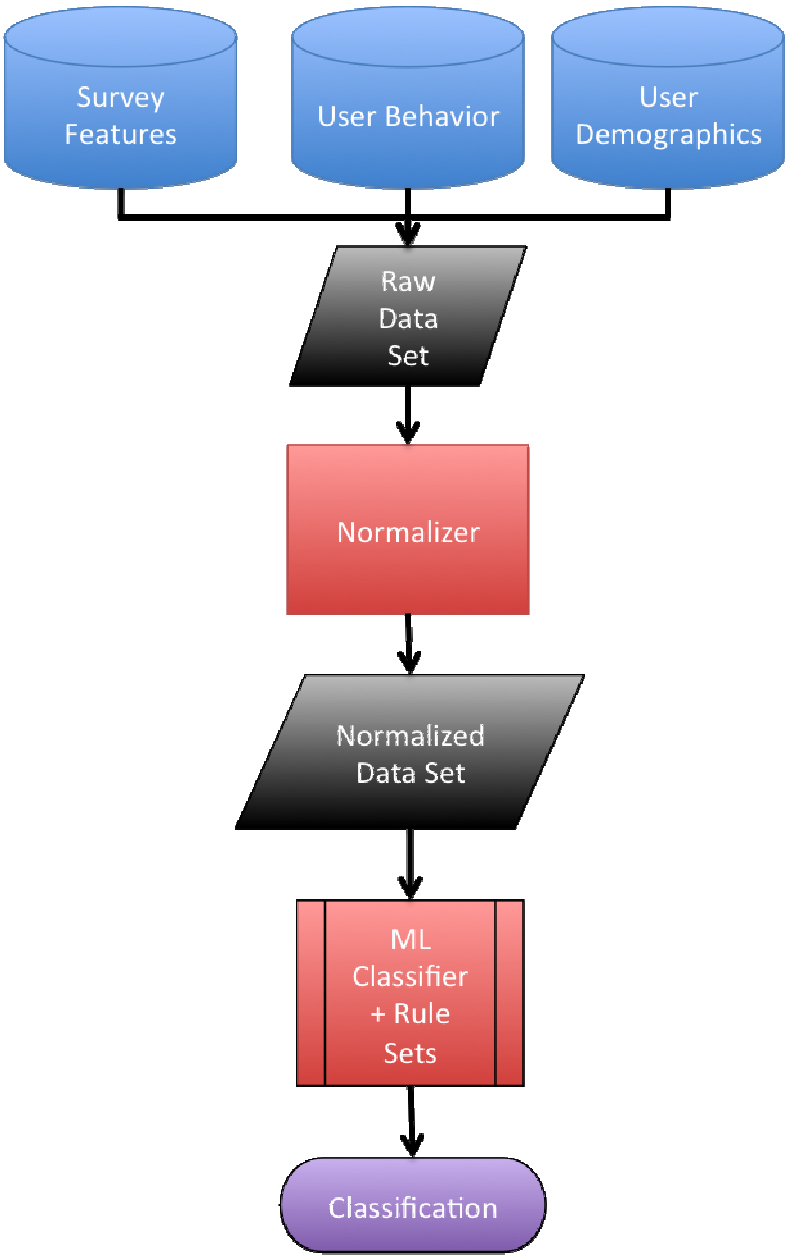

Figure 3.2 – Classification process. To classify data as rare action data or non-rare action data, the data is first gathered from a database of user actions, demographics, and survey features, and then normalized. Following normalization ML techniques are applied to the data, followed by the Rule Set techniques. The result of these

methods is a classification labeling the data as rare action data or non-rare action data.

3.2 Feature Extraction

In addition to developing reliable user models capable of assisting users in web surveys, this thesis extracts features associated with rare actions (i.e. survey error) from the dataset in order to:

1. Provide statistics that we hope will lend guidance to survey methodologists about areas potentially worthy of further research regarding survey design. These researchers are better equipped to meaningfully interpret our dataset, and thus the aim here is to provide initial guidance into research that could lead to survey designs that reduce error (rare action) a priori.
2. Illustrate the potential such datasets have for the survey community as a whole and encourage the use of more informatics and data mining in this field of research. For example, if a system of databases existed across many surveys, methodologists could perform more advanced mining and analysis that could identify trends of survey error on a scale larger and more meaningful than we can accomplish by examining only 1 or 2 surveys at a time. This could lead to more generalizable results, and identify what traits of surveys distinguish one type from another (and thus perhaps classify what kinds of surveys are similar to others). We hope survey methodologists will recognize the value of such databases and begin to pursue the process of building and exploiting such systems.
3. Begin the process of future work in the user modeling problem by identifying which features are potentially the most useful for future models to consider.

For example, we find the number of questions on a page is correlated with breakoff in the Gallup panel, therefore we could make a user model more aggressive to addressing rare actions on such a page.

4. Help inform the generation and refinement of rules since useful and important features can be used to condition the rules. It will also lend insights to domain experts from a data-driven, data-centric perspective.

In order to extract features, simple statistics were derived using the Pearson correlation coefficient, associating features with the amount of rare action in each survey. The features investigated are the same as used by the user model, namely:

- Survey Features
- Respondent Demographics
- Respondent Actions (i.e. behavioral statistics)

From these statistics we can determine which types of users or survey page features are associated with rare action and survey error. This knowledge can then be further investigated in order to determine the cause of these correlations and determine if any dependent relationship actually exists.

Behavioral statistics are based on a view of the entire survey (rather than at some point within a survey). In fact, classifiers using these datasets have very high accuracy for both classes (90-98%) in the Gallup panel but are not usable in practice because their high accuracy rate is due to certain features tipping off the fact that breakoff users did not see every page in the survey. Nevertheless, these datasets can be exploited in order to find what attributes are associated with the rare action of breakoff via correlations.

The survey feature statistics are derived only for Gallup and come from a dataset that gives the amount of error occurring on each page of the survey. Thus we can decorate a page with a description of the number of questions it has, how many words it has, etc. and then investigate whether these statistics are correlated with the amount of breakoff occurring on this page.

The types of features we decorate the pages with are:

1. The page number of the survey – this gives us an idea about whether rare actions increase, or decrease as the survey proceeds.
2. The number of questions on the survey page – pages with more questions are considered more difficult and are thought to affect factors associated with rare action such as motivation, or cognitive ability.
3. The number of words on the survey page – this gives us an idea about the complexity of questions on the page.
4. The Flesch reading ease associated with the text on the survey page – this also gives an idea about question complexity.
5. The number of grid-type questions on the survey page – a grid type question is when one base question is provided and the user must answer this question repeatedly with some part of the question changing in series of sub-questions existing in the "grid". For example, here is a possible grid question: Describe your reaction to the following hamburgers: Big Mac, Whopper, Whataburger, …, etc. Grid questions are known to be associated with error.
6. The number of sub-questions belonging to grids on the survey page. This is similar to the number of grid-questions, but we count the total number of sub-

questions instead. This gives us a better idea of the size of the grids and allows us to investigate if large grids, or very many small grids, are associated with rare action.

7. The number of times the topic changes on a survey page. Topic changes are thought to be unpleasant to survey takers, and thus could affect the underlying causes of rare-actions in surveys.

Furthermore, we do not really know whether the amount of previous pages which should be considered a factor in a user's breakoff detection. Hence, we restrict our viewing to just one page. From this viewpoint, we examine two types of pages – the last page a user answered questions on and the last page they likely saw (the page after the last answer). In this way, we can see if traits of these types of pages are associated with breakoff.

Behavioral and demographic statistics are also investigated for correlations for both ATUS and Gallup. Since there is no common context like pages in an ATUS survey, we do not perform any study analogous to the page characteristics correlations. We do however, attempt to correlate things like the number of time-diary edits to rare-action in ATUS.

These features are valuable to future versions of our model as well as they instruct what environmental features a model should consider when deciding how aggressively to interject itself into the environment. If we know ahead of time that certain pages are more likely to induce a rare action, we can tweak our models in various ways to deal with this fact. For example, using our rule system, we can lower the thresholds in such a way that we expect our identification rate of rare action users to increase (even if the number of

false alarms are raised as well) on pages we expect to display a large amount of rare action. Thus, knowledge of survey features correlated with breakoff empowers our model to be more adaptable to varying costs and benefits on distinct pages; as we may find that we can accept more false alarms on some pages, than on others.

3.3 Rule Examples

This section illustrates some rules and their explanations, which were used to amplify classifier performance and thus enhance our user model. We chose four different rules to serve as examples in this section in order to illustrate the basic ways user modeling rules can be derived from domain knowledge.

Firstly, rules clearly have different natures depending on whether the data is nominal or numeric. Numeric data, in our datasets, often relates concepts such as how often a user performed a task, or the rate of speed at which they perform certain tasks. Nominal data is often related to user demographics, or survey features (such as what type of web browser was used to view the survey). One way nominal and numeric data must be treated differently is in determining what kinds of values can be associated with each other.

For our numeric data, most domain knowledge centers on statements like "younger respondents tend to complete surveys more quickly". As a consequence, in most cases we incorporate such knowledge into the user model by clustering values in some vicinity or at the same extreme end of a spectrum into a rule. This is often how we mimic domain knowledge into rules, in other words. For example, it is known that users who proceed slowly through surveys are theoretically believed to be having cognitive issues. Therefore, we create a rule that states if a user has a question response rate below

a certain boundary-value, then vote that they are a rare-action user. We find the exact boundary-value via exploration. At times, we explore more aggressively by investigating any range of numeric data for a beneficial rule, relying in this case on the domain knowledge that the attribute-value is more generally associated (or deemed relevant to) with form of rare-action.

This strategy is not without some difficulties, however. Note that this strategy exploits numeric data's natural ordering. Whether this ordering is actually valuable depends on the nature of the data we have generated, and what concept this data covers. Some numeric data may be meaningless in its raw form, and only be of value when it is compared to some other value (i.e., normalized). A simple example of this would be a person's weight. The weight alone cannot tell us if someone is over or underweight, the weight information must be normalized based on some other factor such as height, gender or body composition. It is an open question on the best general normalization strategies for the survey behavior data we measure.

We have tried to address the potential dilemma, of using numeric data outside the proper context, via the normalization techniques described in Section 3.1.4. In future work, we will exploit normalization further by creating a new dataset derived from a mix of normalizations based on which normalization approach performs best on a given attribute. We hypothesize this will be a way to improve the user model. The current rule-making technique however, treats each normalization technique agnostically and pretends each dataset has delivered sound measures from which we can deem the order meaningful. Since there is no standard method to perform normalization on survey behavior data, we claim that this is a necessary assumption for the time being; as data

knowledge improves we can refine these techniques.

Note that nominal data has no ordering that naturally exists between the attributes - they are simply meaningful descriptors. There are however, ways in which a person could identify two distinct nominal values belonging to the same set, and distill secondary information valuable to a user model.

Consider color as a nominal attribute. An artist might say that two distinct colors imbue similar effects in a painting or piece of glasswork, and share membership in some set. There exist analogous relationships between nominal attributes in our dataset. This is of consequence as we can create rules with comparison operators such as set membership in our system, and can explore these sets for valuable rules. In this way we can create sets that test for certain meanings not captured by a single nominal value in isolation. For example, testing via rules if the user's operating system is built for a mobile platform can only be accomplished in our system by associating relevant operating system values into a single set. From this set, we can test the hypothesis that mobile user's are more likely to commit rare actions, and create the analogous rule.

Finally, incorporating rules based on domain expertise enhance our user model by allowing the model to lump users into sets known to be associated with the rare actions we detect. In this way, our model better reflects the tendency a user has towards rare actions.

In the following sections we will provide concrete examples of creating nominal and numeric rules in order to illustrate the reasoning further and point out other types of associations that can be made between values.

3.3.1 Numeric Rule Examples

Consider the following rule derived from the Gallup panel for the April, 2012 survey:

- Let s be the amount of scrolls a user performs on a given page normalized to the average user's amount of scrolling.

  If (s >= 5.5) then vote that the case is a breakoff case.

Note that a normalized scroll time of 0 indicates that the user performed no scrolling, and that the higher the value of s, the more scrolling a user performed relative to others. We use frequent scrolling as a proxy for confusion as it indicates user is reexamining, or is having trouble understanding material. Thus, this rule is based on the domain heuristic that more confusion is likely to be associated with more rare actions in surveys (Krosnik, 1991). This rule incorporates that domain knowledge into our user model.

As another example, let n be the number of times a user answered a question on a page, normalized to the average amount of all users. If n is high, it indicates that user is revisiting questions, and if n is low it indicates user is neglecting questions. Generally, the boundary for n indicating of answering vs. not answering questions falls at 1 as most users answer all questions. Somewhat surprisingly, it has been seen that users who take more time answering questions commit fewer survey errors (Fricker et al, 2005) and thus a rule we can make is that if a user has a question answer value, n, above a given threshold, they are unlikely to be a rare action user. In practice for this dataset, we can thus eventually develop a rule that states If (n > 7.2), vote user as non-rare action and thus bring the domain knowledge into the model.

3.3.2 Nominal Rule Examples

Nominal rules are based on the underlying meaning of nominal attributes. One nominal attribute we have for Gallup is the number of computers in the respondent's household; this is relevant as the survey is computer administered (via the web) and thus user's unfamiliarity with the technology may encounter difficulty as survey methodologists often hypothesize about the effects of technology on such users (Dillman and Bowker, 2001). As such, we enhance our user model by flagging the nominal value associated with the presence of no computers as an indication of potential rare action. In practice, if c is the value for household computers, the rule looks like: If (c == value indicating no computers), then vote that user is a rare action user.

All of the nominal rules are derived using domain knowledge, but some are formed of more complex relationships in which multiple values in a set convey a more abstract meaning. We can implement these concepts into the user model, as well. For example, consider marital status. In the Gallup panel, the methodologists try to sign up the entire household into taking the survey. Thus, husbands and wives are often in the panel together and as such may be more motivated to complete the survey. Thus we can attempt to develop rules that associate more abstract forms of marital status, with either rare action or non-rare action. This investigation was performed, and a rule was derived that associates a subset of unmarried people with not having rare actions. Specifically, a rule was developed that states If (the user is divorced or widowed) then vote that the user is normal action user. One effect we might be seeing here, is that older people are more likely to be widowed or divorced, and thus this rule might be simply be restating the fact that older respondents are less likely to breakoff.

Chapter 4

Implementation

As our implementation must mimic our methodology, recall that the methodology we present in this thesis centers on the prediction of rare user action via user modeling. Further, recall that user modeling is a four-step process composed of data collection, data preprocessing, pattern extraction, and system evaluation. In building towards an implementation of a model capable of predicting rare user action we thus had to implement:

1. Data collection mechanisms (discussed in Section 4.1 in Chapter 4)
2. Data preprocessing mechanisms (discussed in Section 4.2 in Chapter 4)
3. Pattern extraction mechanism, i.e. the ML classifiers equipped with a secondary rule technique (the ML classifier implementation is discussed in Section 4.3 and the associated rule technique implementation is discussed in Section 4.4 in Chapter 4)
4. Evaluation mechanisms (discussed briefly in Section 4.5 in Chapter 4, and

Chapter 5)

We discuss each implementation in the following sections.

4.1 Data collection

In Chapter 3, we mentioned that the three sources of data for our user models are respondent behavior (past actions), respondent demographics, and survey features. This section will briefly discuss the exact mechanisms by which data is obtained and stored for each survey. One process linked to data collection is user session recreation, we discuss that task in this section as well.

4.1.1 Data collection and session recreation for the Gallup panel

The Gallup panel data arrives from four sources, which we then feed to a Java based software package (created by our research group called the GallupUtil) responsible for translating the documents into Java objects which are then persisted in a MySQL database also designed by our team. To connect the Java source code to the database we use the Java Persistence API (JPA).

The database is useful as it maintains meaningful relationships between survey components, users, and user demographics, which we can then reflect back into our Java code using the JPA. This eases the construction of meaningful feature sets for user modeling. Additionally, new surveys arrive on a monthly basis, and so a relational database offers an easy way to keep the data updated and consistent.

Specifically, Gallup panel data arrives from four sources:

1. A Microsoft Word document communicating the structure and content of each survey.

2. A Microsoft Excel document containing user responses to each question appearing in the survey.
3. Demographic data, describing each respondent. This information is provided once for each panel member, rather than being provided with each additional survey.
4. Log files, or paradata in survey parlance, describing how the user interacted with the software during the survey. The log files arrive in Microsoft Excel documents.

Translation to Java objects and the database via the GallupUtil is automated for all but the Word document, as we could not determine a known mechanism for parsing such files. Instead, the Word document is first translated by hand to a parsable, structured form. In the future, **we hope to obtain the HTML for the survey representation in addition to the Word document**, as we could then completely automate all processes and remove a source of human error. Interested readers can find a description of the database schema in the Appendix A.

Another important step in data collection for user modeling is session recreation (Frias-Martinez set al, 2006).. The purpose of session recreation is to recapture user experience as it occurred in real-time and extract data from this experience for the user models. To recreate sessions we lean on the log-files (i.e., paradata) as they detail a sequential listing of user actions. These log files afford the opportunity to extract still more data regarding users as well in the form of secondary statistics and behavior.

Whereas one might term a user answering a question, or scrolling within the interface to be a primary action, we can term features such as scroll speed (number of

scrolls / time scrolling) as secondary statistics regarding behavior. Other examples include the number of questions a user skipped on a page, the rate at which a user answers questions, and other actions theoretically indicative of survey error.

One such action is **question response speed**, as "speeding" through surveys is considered a source of measurement error (Guitierrez et al, 2011). Since measurement error and non-response error have similar theoretical causes and sources, tracking such occurrences is valuable to our user modeling. Another example is **straightlining**, wherein a user chooses the same answer choice to a series of multiple choice questions (Guitierrez et al, 2011). We can extract this behavior from the paradata as well. Interested readers can find a complete list of features extracted in Appendix B, as well as a section discussing the theoretical overlap between measurement and non-response error in Chapter 2.

To summarize, we extract these secondary behaviors by examining the log files with a Java program (again part of the GallupUtil software package) and output our findings. This program thus serves as a valuable data collection mechanism for our user modeling.

4.1.2 Data collection and session recreation in ATUS

ATUS data collection also involves several data sources which can be generalized into two categories: public data sets which the Census Bureau provides to the public, and "audit trail files" (i.e., log files, or paradata) that our provided to our research group and describe the usage sessions of actual ATUS interviews. Unlike our work the Gallup panel, we have not yet created a relational database for ATUS and thus rely on a system

of file parsing programs to extract data which we envelope in a Java software package we call the ATUSUtil.

Data extraction from the public data files is rather simple, and involves a Java-based parser capable of translating each file into the appropriate set of Java objects we use to represent the ATUS data. Recreating usage sessions via the log-files is a more difficult, but also more information-rich task. It is also more challenging than the corresponding task in Gallup as the ATUS log files are more granular and noisy, and the current software interface interacts with the underlying program logic in inconsistent, misleading ways that distort the meaning of the log-files. In other words, log-file meaning is dependent on the survey and interface's current states.

In the end, session recreation was accomplished by building a parser that abstracted the paradata lines to changes in survey state. This parser works in tandem with a survey representation that responds to these changes and updates itself based on the communicated state changes and their associated side-effects. One simple example of a side effect is that subsequent activity times are effected by an update to the timeframe of a prior activity.

Furthermore, certain state changes effect the presentation of the current survey software. To mimic this, we utilized the Observer design pattern such that survey state updates were communicated to another software artifact whose role is to track the presentation of the interface and notify its own observers when changes occurred to the presentation's state. In total, we developed a system able to recreate survey sessions in terms of both the state of the survey and the state of the interface any point. We then

gather this data into one place with a tertiary observer watching both the survey state changes and the interface state changes.

We verified the accuracy of this system by comparing the final survey states we generate with the final survey states present in the public data. Ultimately, the system developed is highly functional, and can correctly parse and reconstruct all but 30 of the 13,125 (0.23 %) of the audit trails available in our dataset. To better illustrate the detail of work that must be done to accomplish this kind of data collection for user modeling, we overview the intricacies of this system's design and its associated challenges in Appendix Section D.

The final output of the session recreation program is a set of sequential actions and survey states which occurred during the ATUS session, all encapsulated in Java objects. These actions include the specific value associated with any activity at any point in time, as well as the amount and type of editing that occurred on each activity, and the amount of time spent performing such actions. Since activities can also be inserted into the time diary, and deleted from the time diary, we track these edits as well. A complete description of the information extracted for our user models can be found in the Appendix C. We focused on actions such as editing and time spent editing, as they indicate how the survey is proceeding (i.e. is the time diary proving difficult or easy to create?). We track the current state of activities as the Survey Methodology half our team has found statistical relationships between previous activity types and the memory gap error we are interested in predicting. Thus, these kinds of features were deemed valuable to the user model.

4.2 Data preprocessing

We chose the Weka software package as the basis for our machine learning. Thus one step in pre-processing the data was to translate the Java objects representing user behavior and characteristics into .arff files which Weka uses as input. This was done via Java programs written for both Gallup and ATUS.

Another step in preprocessing is the normalization techniques described in Chapter 3. Each normalization task was implemented by a Java program belonging to either the GallupUtil or ATUSUtil. These normalized data sets were then fed to the .arff file writers described in the preceding paragraph.

4.3 Pattern recognition

In Chapter 3 we communicated that we use basic machine learning classifiers in tandem with the resampling techniques: oversampling, under sampling, Easy Ensemble (Liu et al, 2009), and Balanced Cascade (Liu et al, 2009). Rather than implementing each classifier, we used utilized Weka's machine learning API inside a wrapper software package (that we call the MachineLearningUtil) we wrote to perform experiments and conduct the desired machine learning processes. Since Weka has no implementation of the Balanced Cascade and Easy Ensemble techniques that we know of, we implemented these techniques in Java inside the MachineLearningUtil.

In the end, we developed a Java-based software package capable of performing machine learning experiments as directed by an XML input file. These XML files specify what kind of classifier to use, what kind of resampling (if any to use), what data to test and train on, and what evaluation strategy (for example, cross validation) should be employed. The package is designed to make use of sound software engineering practices, such that if new strategies for classification, resampling, experimental setup, etc. are

desired in the future then we will only have to create new Java classes implementing these features and not have to redesign the package itself.

4.4 Rule implementation

This Java-based software package is also where we developed the rule system. The implementation of the rule system is quite straightforward.

A rule-interface exposes a rule type equipped with one function–to vote on whether the rule thinks an attribute value is an instance of error. In Java, this interface is currently implemented as an abstract class as that allows us to build in a necessary component of all rules into the interface itself, i.e., the attribute type on which a given rule is in charge of basing its votes.

A rule-set interface exposes a type equipped with a function which takes an entire data instance as input and outputs the rule-set's classification of the instance; that is, the instance is classified as being a case of rare action or not rare action. Behind the scenes, each implementation of the interface encapsulates a set of rules and determines how to combine them in order to generate the final classification. We implement this interface as a true Java interface rather than an abstract class.

Behind these interfaces, we build actual implementations of rules and rule sets. The rules are all extensions of the Rule abstract class and vary only in the function (i.e. comparison operator) they use to evaluate the attribute type they are tasked with basing their classifications on. Recall from Chapter 3 that there are multiple comparison operators we have implemented, and that these can be expanded in future work.

Implementations of the rule set interface vary only in the mechanism by which they combine rules. For the experiments conducted in this thesis only one concrete

implementation was used—a combination strategy of raw voting. That is, if more than x rules vote that a instance is of a given class, then that is the class to which the rule-set assigns the instance. Alternatively, as a research group, we have envisioned future rule set implementations that combine rules based on a weighting scheme such that rules proven to be more reliable, or deemed more reliable in a given context, are given more weight in the voting than their counterparts. In future work we can explore these different strategies.

Currently though, we explore our implementations by changing the number of rules that must vote for a class before an instance is assigned to that class. Recall from Chapter 3 that we call this number of votes the threshold.

Having described the implementation of the rules themselves, we now describe the implementation used to find and extract the rules. Recall in the methodology section that we discussed two basic ways attributes can be categorized—nominal and numeric—and that each of these categories has a different mechanism by which we search for rules.

To reiterate, we look for nominal rules in pre-coded value sets that convey an abstract meaning when the values are placed into the same set (such as a set of operating system values each of which is associated with mobile device). To implement this search, we simply put the combinations we are interested in into files that a Java program then parses for its basis of rule search.

Numeric attributes are searched in terms of ranges. This requires no file input. Instead we can find the ranges to search by using simple loops which iterate over a set of realistic values we could find for each attribute. As these loops are iterated, we output potential rules by associating the current boundary value we occupy in the loop with one

of the operators we plan on exploring. For example, if the current value in the loop is 100, we can output of a rule of the form **attribute-value < 100**, hence 100 is the "boundary-value" for the rule.

After generation of the potential rules we, evaluate each rule in isolation by examining its performance on a set of user-session data. We evaluate by examining how frequently each rule overturns the current classifier's classification on the training data, and how often this overturning is correct. We then select the best rules for each attribute type, operator pair whose use positively impacts the geometric mean of the original classifier. This is done via a Java program which examines both the dataset and the output of the base classifier originally used to classify the data.

The final step in rule searching is to **amalgamate** the rules into a pair of final rule-sets. Recall, one set is generated from rules whose role is to overturn rare action classifications, and one rule set is generated to overturn non-rare action classifications. This amalgamation is done with a Java program that examines the best operator-attribute value rule pairs for each attribute, and determines which pair to use. The amalgamation process also places each selected pair into the appropriate rule-set, i.e., the set turning rare action cases to non-rare action or the opposite set. Importantly, the **amalgamator** will combine operator-attribute value pairs with distinct operators into a single rule encapsulating both rules if the two rules are logically consistent and distinct. For example, if we find that a sound rule is respondents under 30 are prone to the error of interest and respondents over 90 are also prone to the error, we amalgamate the two rules into a single rule of the form respondents under 30 and over 90 are prone to breakoff.

In this way we generate rule sets likely to be reliable to the user model.

To combine the rules with the classifiers, we utilize our Java software package containing the Weka wrapper's and the rule implementation. We read in the rules from files generated to represent the rule-sets derived during rule amalgamation. We then use the rule evaluation and classifiers together to classify data with the classier first classifying the data and the appropriate rule-set then possibly overriding the decision.

4.5 Evaluation implementation

To evaluate our model we utilized the Holland Computing Center's Tusker system. Evaluation consisted mainly of creating bash-scripts responsible for generating, storing, and passing the appropriate input and output files to each component in our system.

Chapter 5

Results

The methodology regarding building user models for predicting rare actions centers on two points: the use of resampling techniques in conjunction with ML classifiers, and then appending these classifiers with a domain-knowledge based rule technique used to override classifications and improve performance.

As such, the studies presented in this Chapter pursue the following series of objectives:

1. **Objective 1**: Demonstrate that ML classifiers equipped with resampling techniques are a potential approach to identifying rare action users in an environment characterized by a lack of self-direction, limited content navigation, and presence of restricted content. We show this by demonstrating that the techniques predict rare action users at a rate significantly better than random guessing.

2. **Objective 2**: Demonstrate that domain knowledge applied to the user modeling classifiers improves performance. We show this by hand-crafting sets of rules based on domain knowledge for twenty-four data sets, and demonstrate that the performance is significantly better the technique with no domain knowledge.
3. **Objective 3**: Demonstrate that extracting rule-sets through a mixture of domain knowledge and data exploration succeeds in improving classifier performance. We argue that this is more feasible and realistic than the technique illustrated by Objective 2.
4. **Objective 4**: Demonstrate that automatic extraction of rule sets provides our user models with a repeatable, generalizable method of applying domain heuristics to the improvement of user classification as rare-actors.
5. **Objective 5**: Show that the extracting rule-sets methodology can boost detection of rare-actions in multiple settings, by performing experiments with rule equipped classifiers and non-rule equipped classifiers in ATUS.
6. **Objective 6**: Demonstrate that rare-action centered user-modeling in ATUS is feasible, by proving that user-session data and demographic information can detect rare-actors at a rate significantly better than random.

The results of the studies will show how the rule-set methodology boosts classifier performance and how we are able to predict rare-user actions in the hostile user modeling environments associated with surveys. Each study was performed on the Holland Computing Center's Tusker system using the projects mentioned in Chapter 4.

Another set of objectives addressed by the experiments in this Chapter relates to the extraction of software and user experiential features associated with rare-actions. We investigate these relationships with the following objectives:

1. **Objective 7**: Determine if any environmental features are associated with the presence of rare-action in the Gallup panel.
2. **Objective 8**: Determine if rare-actions are associated with certain trends in user-behavior and certain user-demographics.

The basis for investigating Objectives 7 and 8 is the Pearson correlation coefficient. The Pearson correlation coefficient is used as it can assess how related the presence of certain contexts are to user rare actions, and thus provides a good source of initial information regarding design considerations and contextual traits effective user-models should pay heed to. The Pearson correlation coefficient is also used in the context of more behavioral statistics regarding user actions. Learning about the correlations of these statistics to rare-action grants insight into whether certain behavioral features can be used as strong indicators of user's tendency to rare action. This informs both present and future work. This is particularly interesting if we derive this statistic from whole measures taken after completed surveys, as the behavioral statistics derivable after a session has expired offer a complete picture of user-session not available in real-time. Thus, we can observe trends in user-rare actions, such as whether Gallup users tend to commit rare-actions at early or late in surveys by observing if a statistic like total user session time has correlation in regard to rare-actions.

Before discussing each study in Sections 5.3, 5.4, 5.5, 5.6, 5.7, and 5.8, we will discuss some background information regarding the Matthew's Correlation Coefficient in

Section 5.1; a statistic which can be used to determine how well a ML classifiers predictions agree with the actual values of data. This statistic was chosen as it offers a "more balanced evaluation" of classifier performance than other options such as percentages based on the amount of true positives, false positives, and other confusion matrix members (Baldi et al, 2000); it is therefore more appropriate for assessing classifiers on our imbalanced data sets.

We will then discuss the Wilcoxon Signed Rank test, which we use to compare the performance of rule-set employing predictive models against classifiers not using rule-sets, in Section 5.2. We utilize the Wilcoxon Signed Rank test as it is the recommend method of comparing the performance of two classifiers on multiple, independent data sets (Demšar, 2006) and so, given the way we compare the rule-set equipped classification against the lone classifiers, is the most appropriate choice.

Since the Wilcoxon Signed Rank Test compares to classifiers over multiple data sets, we must select a statistic on which to base our comparisons. We use the geometric mean (g-mean) of the true positive rate and true negative rate as this basis as this is a statistic insensitive to class imbalance that is recommended for use in such situations (He and Garcia, 2009); this was discussed previously in Chapter 2.

In Section 5.9 we will summarize our findings and place them back into the context of the traits distinguishing the user modeling problem in surveys, from those of the traditional user modeling applications: one in which users possess: a lack of self-direction, a limited ability to navigate content, and a restricted content set that all users must engage with.

5.1 Matthew's Correlation Coefficient and Chi-squared test

Recall that in Chapter 2 we communicated the fact that precision is a statistic sensitive to class balance and as such is a statistic difficult to improve in imbalanced data sets. We gave as an example, a typical Gallup data set composed of 17,000 users with approximately 300 rare-action users and pointed out that a precision on the rare action users of 30% could imply a performance such as 100% recall on rare action cases and 95% accuracy on non-rare action cases. As such, we mentioned we would shift discussion of results away from precision and towards statistics that are not sensitive to class balance such as recall and the geometric mean of the true positive rate and true negative rate (where rare action cases are considered positives, and non-rare action cases are considered negatives).

Another such statistic is the Matthew's Correlation Coefficient (MCC). The MCC is essentially the Pearson correlation statistic applied to assessing the dependence between the output of a ML classifier and the actual value of the data (Baldi et al, 2000). A value of +1 indicates that the ML classifier completely agrees with the actual values, a value or -1 indicates the ML classifier completely disagrees with the actual values, and a value of 0 indicates the ML classifier is making completely random predictions (Baldi et al, 2000).

The MCC can be computed from the number of True Positives (TP), False Positives (FP), True Negatives (TN), and False Negatives (FN), by the following formula:

$$\frac{TP \times TN - FP \times FN}{\sqrt{\left((TP+FN)(TP+FP)(TN+FN)(TN+FP)\right)}}$$

The MCC is also related to the Chi-squared statistic, χ^2, by the equation, where N is the size of the data set:

$$\chi^2 = N \times MCC^2$$

From this equation we can derive the Chi-squared statistic for each MCC and use it to perform a Chi-squared test for a 2 x 2 contingency table to determine if the MCC is significantly different than 0 (Baldi et al, 2000). Successfully passing this test indicates that a classifier performs statistically better than random guessing.

For the tests on the Gallup data sets we compute the MCC and its associated Chi-squared statistic. We treat rare-action cases as positives, and non-rare action cases as negatives.

5.2 Wilcoxon Signed Rank Test

The studies by which we compare the performance of classifiers with rule-sets against classifiers without rule sets are done across a series of independent data sets. The Wilcoxon signed rank test is the recommended technique by which to compare two classifier's performance over multiple datasets (Demšar, 2006). As such, we use it to compare the performance of our rule-set and classifier two-tier technique against the performance of a 1-tiered classifier.

The basic idea of the test is to first compute the performance difference of two classifiers on each data set. These differences are then sorted in ascending order according to their absolute values. Each difference in the sorted list is then assigned a rank starting at 1, for the smallest difference, and incrementing by 1 at each step.

The test works as follows for two classifiers C_1 and C_2 (Demšar, 2006); for a one-sided test, the null hypothesis (i.e. the condition the test is attempting to disprove via the

test) is that C_1 performs no better than C_2, and the alternative hypothesis is that C_1 outperforms C_2:

1. First, the differences in performance on each independent data set are computed, by subtracting the performance of classifier C_2 from the performance of classifier C_1.
2. The absolute values of the differences are then ranked in ascending order, starting with one. If there are any ties, each difference receives the average of the rankings the differences would span if they were different. Simultaneously, the differences are placed into three groups: one group contains the differences whose value was positive, one group contains the differences whose value was negative, and one group contains the differences of value zero. We can thus note what sign each ranked test result has: positive, negative, or zero.
3. Two statistics are then computed, one indicating the sum of the ranks of the positive differences, R^+; and one indicating the sum of the ranks of the negative differences, R^-. The ranks of the zero-differenced results are split evenly between R^+ and R.
4. The lesser of the two statistics, R^+ and R^-, is used as the Wilcoxon test statistic, *T*.
5. *T* can then be converted to standard score by the formula below, or by looking it up in a table based on the number of paired tests, *N*:

$$z = \frac{T - \frac{1}{4}N(N+1)}{\sqrt{\left(\frac{1}{24}N(N+1)(2N+1)\right)}}$$

6. If $z < -1.96$, we can reject the null hypothesis that there is no difference in performance at the 97.5% confidence level.

We apply this procedure to assessing the difference in performance between rule set classification and non-rule set classification.

5.3 ML classifiers and resampling techniques in predicting rare user actions

The first experiment we present is a study of the performance of ML classifiers and resampling techniques in terms of identifying users as rare-actors or non-rare actors.

Our objective in this experiment is the following:

Objective 1: Demonstrate that ML classifiers equipped with resampling techniques are a potential approach to identifying rare action users in an environment characterized by a lack of self-direction, limited content navigation, and presence of restricted content. We show this by demonstrating that the techniques predict rare action users at a rate significantly better than random guessing.

5.3.1 Setup

To restate what was mentioned in Chapter 3, we isolate Gallup data on a page by page basis, and can use this data to build a sequence of classifiers for categorizing users as rare vs. non rare actors. We also investigate four normalization techniques to the Gallup data: normalization against the average user behavior, normalization of the user's current behavior to their own average behavior on past pages, basing the user behavioral measures based on which unique page they are currently interacting with, and by basing the measures on which page number of interaction they are currently on (thus after a user

interacts with 3 pages, their interaction data will be logged in a dataset that is associated with each user's 3rd page of interaction).

The combinations of resampling, classification, and number of datasets lead to many possible experiments. For instance, we explore five ways of resampling the data (oversampling, undersampling, no resampling, Easy Ensemble resampling, and Balanced Cascade resampling), the use of three base classifiers (naïve Bayes, decision trees, artificial neural networks), the use of four normalization techniques, and there are approximately 20 pages on each survey (recall that in Gallup each normalization technique is centered on building a dataset for each page, and consequently a classifier for each page). Rather than launching each possible experiment, we chose to focus our investigation on pages known to have a high incidence of breakoff (which we determined by counting the amount of breakoff occurring on each unique page and after each possible amount of pages a user had seen) as well as a few random pages from each month. Therefore, for each month we chose the top 2-4 pages in terms of breakoff incidence (2-4 as the amount of pages with breakoff obviously much larger than the average varies from survey to survey) along with 2-4 random pages for a total of 6 pages on each survey. We then evaluated each classifier's performance using each of the resampling techniques (as well as performing no resampling) for a total of 432 total tests. The basis of the tests was 10-fold cross validation.

5.3.2 Results

Even this is a huge amount of data to sift through. And so to evaluate the feasibility of our techniques we focused our attention on the set of 24 unique normalization-month combinations that yielded the highest g-means compared to their counterpart

normalization-month combinations. That is we selected the top normalized to self result from each month, for instance, and did the same for the other normalization techniques. This was done to get an idea about the ceiling of performance classification and resampling has using our current methodology.

We did not engage in a comparative study amongst resampling techniques, as this was not deemed necessary to this study's objective, but do note that each of the top results employs either Balanced Cascade or Easy Ensemble resampling. The top results are presented in Table 5.1, below. We show the recall, g-mean, and the MCC and its related Chi-squared statistic. Note that the MCC's Chi-squared statistic indicates that each result achieved classification at a rate significantly better than random guessing. This indicates that applying user modeling to the unique environment surveys offer is feasible.

The key for Table 5.1 appears below:

Data Set – the data set which was tested. These are identified by the month of the survey and the page of the survey. The page of the survey can be either the page number from the user's perspective, or the unique page identification number that the page has been assigned out of the entire field of pages. The former applies to the Page In Order normalization technique, and the latter applies to the other normalization techniques.

Norm. Technique – the normalization technique used to create the data set.

Resampling – the resampling technique applied to the data.

Classifier – the ML classifier used in conjunction with the resampling

Rec. – the recall in terms of performance on identifying rare-action users.

Prec. – the precision in terms of performance on identifying rare-action users. Note that the precision is quite poor due to the class balance problem.

True neg. rate – the true negative rate in terms of identifying non-rare action users (true negative / (true negative + false positive)).

G-mean – the geometric mean of the recall (true positive rate) and true negative rate. Represents how well on average the classifier performs on identifying users.

MCC – the Matthew's Correlation Coefficient. An MCC of 0 indicates that a classifier performs no better than random guessing.

χ^2 – the Chi-squared statistic associated with the MCC. The Chi-squared value informs us if the MCC statistic is significantly different than an MCC of 0. The Chi-squared value we must surpass to achieve this is 7.789 at the 99.5% confidence level.

Data Set	Norm. Technique	Resampling	Classifier	Rec.	Prec.	True Neg. Rate	G-mean	MCC	χ^2
January - Page 2	Page In Order	Easy Ensemble	Decision Tree	0.636	0.047	0.640	0.638	0.093	155
June - Page 10	Page In Order	Easy Ensemble	Decision Tree	0.640	0.021	0.628	0.634	0.061	71
April - Page 4	Page In Order	Balanced Cascade	Decision Tree	0.656	0.018	0.648	0.652	0.063	78
September - Page 13	Page In Order	Balanced Cascade	Naïve Bayes	0.814	0.008	0.753	0.783	0.064	74
November - Page 5	Page In Order	Easy Ensemble	Decision Tree	0.612	0.021	0.646	0.629	0.058	59
December - Page 9	Page In Order	Balanced Cascade	Decision Tree	0.684	0.012	0.679	0.681	0.058	59
January Unique Page 106	Normalized To Self	Easy Ensemble	Decision Tree	0.741	0.027	0.680	0.710	0.097	275
April - Unique Page 5	Normalized To Self	Easy Ensemble	Decision Tree	0.721	0.020	0.701	0.711	0.085	169
June - Unique Page 30	Normalized To Self	Balanced Cascade	Decision Tree	0.642	0.015	0.654	0.648	0.056	31
September - Unique Page 45	Normalized To Self	Easy Ensemble	Naïve Bayes	0.688	0.009	0.722	0.704	0.054	53
November Unique Page 56	Normalized To Self	Balanced Cascade	Decision Tree	0.614	0.019	0.640	0.627	0.056	54
December - Unique Page 86	Normalized To Self	Balanced Cascade	Decision Tree	0.647	0.007	0.642	0.644	0.038	25

January - Unique Page 99	Normalized To Average User	Easy Ensemble	Decision Tree	0.620	0.037	0.640	0.630	0.079	113
April - Unique Page 5	Normalized To Average User	Balanced Cascade	Decision Tree	0.716	0.020	0.701	0.709	0.084	165
June - Unique Page 25	Normalized To Average User	Easy Ensemble	Decision Tree	0.622	0.022	0.649	0.635	0.063	78
September - Unique Page 45	Normalized To Average User	Balanced Cascade	Decision Tree	0.719	0.008	0.682	0.700	0.051	47
November - Unique Page 55	Normalized To Average User	Balanced Cascade	Decision Tree	0.631	0.026	0.631	0.631	0.067	78
December - Unique Page 75	Normalized To Average User	Balanced Cascade	Decision Tree	0.910	0.059	0.761	0.832	0.195	1338
January - Unique Page 106	Normalized By Unique page	Easy Ensemble	Naïve Bayes	0.667	0.006	0.661	0.664	0.040	29
April - Unique Page 6	Normalized By Unique page	Balanced Cascade	Decision Tree	0.697	0.021	0.722	0.709	0.086	173
June - Unique Page 32	Normalized By Unique page	Balanced Cascade	Decision Tree	0.725	0.004	0.656	0.690	0.037	26
September - Unique Page 44	Normalized By Unique page	Balanced Cascade	Decision Tree	0.747	0.013	0.724	0.735	0.072	96
November - Unique Page 55	Normalized By Unique page	Balanced Cascade	Decision Tree	0.646	0.010	0.641	0.643	0.044	34
December - Unique Page 76	Normalized By Unique page	Easy Ensemble	Decision Tree	0.610	0.025	0.631	0.620	0.061	68

Table 5.1. Best results for each month-normalization combination, in terms of g-mean, when using ML classifiers with resampling techniques to predict rare-user actions.

5.3.3 Discussion

The results show that the **current ceiling of classifier performance is above random guessing and thus that utilizing user models in an environment characterized by a lack of self-direction, limited content navigation, and presence of restricted content**, **is a feasible enterprise**. A more complete presentation would include results for each dataset tested, but we did not deem this worth the space of discussion here. Here we simply wish to show that resampling classifier application to user rare action identification in the unique environment found in surveys is a potential avenue to explore,

and to find a set of classifiers and data sets with which we can compare non-rule using results to rule-using results. That is one reason we focused on the "ceiling" of performance. In the two studies that follow, we will reuse these ceiling results to compare the performance of rule-based classifiers to those not using rules.

In conclusion, we claim that we were successful in regards to Objective 1, and that classifiers combined with resampling techniques can be used to identify rare action user's at a rate significantly better than random guessing in the environment affiliated with surveys. However, the current ceiling of results still leaves a lot of room for improvement, especially in regards to the ratio between the rare-action cases identified and the non-rare action cases misidentified as rare—i.e., true positive over false positive numbers. While the class balance issue does have a large effect in explaining the poor precision, even in a balanced data set we would be facing a precision of approximately 65-72% in most of these data sets as our geometric mean is typically in this region, and the true negative rate and recall are typically about equal.

Thus, though this scheme is better than random, there is still much room for improvement. The recall as well, which is more resilient against class balance issues, warrants further investigation for improvement as it again remains in the 65-72% range generally. Hence, even if we are unable to reduce the false positive rate significantly, addressing the recall issues could be a valuable pursuit.

These considerations motivate investigations aimed at addressing these performance issues. In this thesis, we pursued this task by injecting domain knowledge into the user-modeling classifiers via the rule-sets outlined in the Methodology. In the

subsequent sections, we will show that this technique does improve performance at a statistically significant level.

Future investigations too are motivated by these results. For example, other ML techniques, such as Sequential algorithms, could be applied to the task of user-classification that might yield superior performance in terms of the geometric mean, and consequently, the recall on rare-action users and the false positive rate (positive being rare action users) as well. As a brief aside, preliminary studies were in fact conducted using long short term memory neural networks, but the results were so poor (0% accuracy on rare action cases for the Gallup panel, most notably) that we shifted to the strategy presented above of combining resampling with non-sequential classifiers. It is possible however, that a different sequential algorithm or another tactic, could yield better results.

5.4 Demonstrating the potential of domain knowledge

The first explorations we did using rules were to generate rule-sets by hand, based on the domain-knowledge in the literature. The associated experiments had the following objective:

> **Objective 2**: Demonstrate that domain knowledge applied to the user modeling classifiers improves performance. We show this by hand-crafting sets of rules based on domain knowledge for five data sets, and present that the performance is significantly better the technique with no domain knowledge.

5.4.1 Setup

To determine whether hand-made, domain knowledge based rule-sets could improve user rare-action prediction, we performed 24 pairwise tests. The data sets used represented

four data sets from each month (April, June, September, November, December, January). The four data sets from each month represented each normalization technique applied to one page of the survey's data. Since, as in the experiments outlined in Section 5.3, we applied each normalization technique to six survey page data sets in each month and using each of the 5 resampling techniques, we still had thirty datasets for each month-normalization combination to choose from in determining which exact 24 datasets to use.

Rather than use all of the results, we chose the classifier-resampling-dataset combination that performed best amongst these 30 candidates. We did this, as we felt it represented a tough-standard against which to compare the rule-based techniques.

After selecting the 24 dataset and associated classifier-resampling techniques we constructed a True-Negative Rule Set and True-Positive Rule Set for each normalization technique by hand, basing each one on domain knowledge. We then equipped the user-classification scheme associated with each data set with the appropriate pair of rule sets. Put another way, each of the six normalized-to-self data sets (one from each month) received the same pair of rule-sets; likewise for the other normalization techniques.

In determining the rules, first domain knowledge was exacted from the literature, such as older users being less likely to breakoff, and then specific boundaries had to be determined for each rule. This is the tricky part of crafting rules by hand, especially given that we normalize the data and thus distort values such as the number of times a user scrolled with their mouse, or answered a question on a given page. Hence, the only way we had to come up with reasonable boundaries to examine a dataset using the same normalization technique and determine what a reasonable number would be. This was done by binning values into sets of 10 in order to get an idea about what boundary values

represented the demarcations of “extreme-values” such as the top 10% of question answerers.

Examining the data this way is an obvious weakness of the hand-crafted rules for at least two reasons. One, we should be rightly skeptical that boundaries can be consistent across pages as the context of the user’s environment changes. Two, one could rightly question whether examining the data to base the rules on violates the conditions for a valid experiment, one being to test classifications on unseen data. We agree with these assessments and are motivated by them to perform the automated rule construction techniques outlined in Section 5.5. We also, in regards to this experiment, would point out that in order to find boundary values in the face of normalized data have to examine some data in order to get a realistic boundary value that actually represents domain knowledge such as “user’s answering more questions may be having cognitive difficulty”. From this perspective, the only way we can test the utility of domain-based rules is to actually represent them properly, and the only way to do that is to examine some data. Secondly, the rule sets were created from examining a single data set and then applying the constructed rule-set to other, unexamined, data-sets employing the same data scheme. All that being said, we would not recommend using hand-crafted rules centered on user-behavior (though we might for demographics such as age) in practice for both the two complaints above and for other reasons of practicality which we will discuss following presentation of the results. In summary, we engage in the study of hand-crafted rules primarily as a matter of proof-of-concept that domain knowledge rules have utility, and as motivation for work which constructs a generalizable method of

extracting and applying domain based rules but would not recommend a system based solely on hand-crafted rules in practice.

Table 5.2, below on the next page, enumerates the rules created for each normalization technique and explains each rule's roots. As a reminder, the role of the True Negative Rule Set is to convert rare action classifications to non-rare action classifications, and the role of the True Positive Rule Set is to convert non-rare action classifications to rare-action classifications.

Note that each attribute is assigned at most one rule, and that we have ranged the size of the rule-sets from one at the smallest, to nine at the largest. Each rule is presented in the form of the domain knowledge in the table, with the precise specifications for the rule in terms of attribute name, comparison operator, and boundaries appearing in parentheses beside it.

To generate any rule by hand we must come up with a way of projecting the domain knowledge, which is often fuzzy, onto a concrete rule. This is at times difficult as such choices are fuzzy by their very nature. In general the process requires first projecting the domain knowledge onto an abstract concept, and then fitting an attribute to the concept. For example, we project the domain knowledge unmotivated users are more likely to commit survey errors (and hence, the rare-action of breakoff) by creating a rule regarding users skipping questions. The concrete form of this rule is that if users answer less than 85% of the questions they view, to consider them potential rare actors. This rule appears for the True Positive Rule Set of the Normalized By Page Order data sets. To arrive at this concrete rule, we must first project the domain knowledge onto an appropriate concept, in this we project motivation onto user skips as not answering

questions can be attributed to motivation, and then project the concept onto an appropriate attribute – in this case percent of questions answered, a suitable way to measure skips.

Other rule derivations are more direct. The simplest example is the domain knowledge that older users are less likely to breakoff. This can be directly applied to our attributes as we have a precise representation of age. In most cases however, some projection is required.

Normalization Technique applied to Data Set	**True Negative Rule Set**	**True Positive Rule Set**
Normalized-to-self	Respondent is older than most users (age > 70)[1]	Respondent is younger than most users (age <= 38)[2]
Normalized-to-average user behavior	Respondent is older than most users (age > 70) [1] The user's technology is behaving correctly (time taken to download new survey pages normalized to the average time for users < 8)[3] The user is careful about answering questions (number of questions answered normalized to	User is younger than most other users (age <= 35)[2] The user is unfamiliar with the technology used to conduct their survey session (computers in house = 0)[5]

[1] Older respondents are less likely to quit surveys before finishing (Peytchev, 2009)
[2] Younger respondents are more likely to quit surveys before finishing (Peytchev, 2009)
[3] Technical difficulties increase the likelihood of users becoming frustrated, and the chance of the user losing motivation (Peytchev, 2009). Therefore, if we don't observe technical problems rare-actions (i.e. survey error) is less likely.

	the average amount for users > 2.25)[4]	
Normalized by unique page	Respondent is older than most users (age >= 75)[1]	Respondent is younger than most users (age <= 43)[2] The user scrolls much more frequently than a typical user, indicating they are confused or unhappy with the current questions (the number of scrolls the user performs per page > 16)[6]
Normalized by page order	Respondent is older than most users (user age is > 75)[1]	The user answers more questions per page than typical users (normalized question responses per page > 2.9)[7] The user is skipping questions frequently (percent of questions answered < .85)[8] The user is spending a lot of time reviewing question text (average time per scroll > 2.5 seconds)[9] The user is rushing through questions (average question response time < 3.2)[10]

[5] Unfamiliarity with technology can increase the difficulty of the survey task (Dillman and Bowker, 2001)
[4] Users more carefully answering questions are less likely to commit survey errors as they are likely more motivated (Krosnick, 1991)
[6] Confusion on questions, or finding current questions unpleasant ties is triggered by problems with motivation, or question difficulty increasing the chances of other survey errors (Krosnick, 1991)
[7] This indicates the user is likely re-answering questions, indicating they may find the questions difficult, increasing error chances (Krosnick, 1991)
[8] Skipping questions shows that the user is likely not motivated to answer questions (Bosnjak and Tuten, 2001). Less motivated users are more likely to commit errors and thus engage in rare actions
[9] Spending long amounts of time reading text or answering questions, can indicate the user is having trouble understanding the questions (Yan and Tourangeau, 2008), increasing the odds of rare-actions through cognitive difficulty problems.
[10] Recall from Chapter 2 that motivation is linked to survey errors, and hence rare-actions.

		or is struggling to answer questions (average question response time is > 11 seconds)[9] The user is struggling to answer questions (question response time per page > 113 seconds)[9] Average time gap between answering last question to requesting next page is faster than the typical user, potentially indicating motivational issues[10], or speeding[11] (average page download time < 4.5 seconds) Total time spent per page is longer than typical users (time per page > 99 seconds)[9] User's daily internet usage is less than typical users (internet usage is less than once a week, once a week, or never)[12] User's device is a tablet or phone[13]

Table 5.2. Prescribed rules and explanations

Also, note that due to the projection process there are multiple ways to represent the same piece of domain knowledge. A good example is the user's experience with technology, which we can represent through either their Internet usage, or presence of household computers. Another example is fitting motivational problems onto rules. In the

[11] Speeding is a form of measurement error (Guittierez et al, 2011) and, as shown in Chapter 2, measurement errors have the same underlying causes as other survey errors like breakoff

[12] User's familiar with internet technology have an easier time engaging with web-surveys (Yan and Tourangeau, 2008)

[13] The web surveys are designed for desktop or laptop computers. Hence, users using a mobile device can encounter difficulties, giving them less "opportunity" to answer questions properly increasing error chances (Krosnick, 1991).

rules in Table 5.2 we do this in multiple ways: the amount of time answering questions and the user skipping questions, for instance.

5.4.2 Results

We present a table of results below in Table 5.3 (the key is the same as that of Table 5.1). Note that each result performs better than random guessing as indicated by the MCC and associated Chi-squared statistic. In this study, we are more interested in comparing rule-set performance against non-rule set classifier performance. To do this, we performed the Wilcoxon Signed rank test.

Rules or No Rules	Data Set	Norm Tech-nique	Resam-pling	Classi-fier	Rec.	Prec.	True Neg. Rate	G-mean	MCC	χ^2
No Rules	Jan-uary - Page 2	Page In Order	Easy Ensem-ble	Decision Tree	0.6364	0.0472	0.6405	0.6384	0.0935	155
Rules	Jan-uary - Page 2	Page In Order	Easy Ensem-ble	Decision Tree	0.6364	0.0472	0.6402	0.6383	0.0934	155
No Rules	June - Page 10	Page In Order	Easy Ensem-ble	Decision Tree	0.6398	0.0208	0.6277	0.6337	0.0607	71
Rules	June - Page 10	Page In Order	Easy Ensem-ble	Decision Tree	0.6441	0.0207	0.6241	0.6340	0.0607	71
No Rules	April - Page 4	Page In Order	Balanced Cascade	Decision Tree	0.6564	0.0180	0.6485	0.6524	0.0626	78
Rules	April - Page 4	Page In Order	Balanced Cascade	Decision Tree	0.6923	0.0182	0.6322	0.6616	0.0660	87
No Rules	Sep-tem-ber - Page 13	Page In Order	Balanced Cascade	Naïve Bayes	0.8140	0.0078	0.7529	0.7828	0.0637	74
Rules	Sep-tem-ber - Page 13	Page In Order	Balanced Cascade	Naïve Bayes	0.8140	0.0083	0.7701	0.7917	0.0673	82
No	No-	Page	Easy	Decision	0.6117	0.0205	0.6459	0.6285	0.0585	59

Rules	vem-ber - Page 5	In Order	Ensem-ble	Tree						
Rules	No-vem-ber - Page 5	Page In Order	Easy Ensem-ble	Decision Tree	0.6602	0.0199	0.6056	0.6323	0.0591	60
No Rules	De-cem-ber - Page 9	Page In Order	Balanced Cascade	Decision Tree	0.6837	0.0119	0.6792	0.6814	0.0580	59
Rules	De-cem-ber - Page 9	Page In Order	Balanced Cascade	Decision Tree	0.6837	0.0121	0.6841	0.6839	0.0590	61
No Rules	Janu-ary Uniqu e Page 106	Nor-ma-lized To Self	Easy Ensem-ble	Decision Tree	0.7413	0.0269	0.6803	0.7101	0.0972	275
Rules	Janu-ary Uniqu e Page 106	Nor-ma-lized To Self	Easy Ensem-ble	Decision Tree	0.7703	0.0254	0.6467	0.7058	0.0939	257
No Rules	April - Uniqu e Page 5	Nor-ma-lized To Self	Easy Ensembl e	Decision Tree	0.7214	0.0204	0.7012	0.7112	0.0848	169
Rules	April - Uniqu e Page 5	Nor-ma-lized To Self	Easy Ensem-ble	Decision Tree	0.7114	0.0209	0.7124	0.7119	0.0860	173
No Rules	June - Uniqu e Page 30	Nor-ma-lized To Self	Balanced Cascade	Decision Tree	0.6420	0.0153	0.6543	0.6481	0.0565	31
Rules	June - Uniqu e Page 30	Nor-ma-lized To Self	Balanced Cascade	Decision Tree	0.6790	0.0135	0.5843	0.6299	0.0485	23
No Rules	Sep-tem-ber - Uniqu e	Nor-ma-lized To Self	Easy Ensem-ble	Naïve Bayes	0.6875	0.0086	0.7218	0.7044	0.0538	53

	Page 45									
Rules	September - Unique Page 45	Normalized To Self	Easy Ensemble	Naïve Bayes	0.6875	0.0095	0.7488	0.7175	0.0592	64
No Rules	November Unique Page 56	Normalized To Self	Balanced Cascade	Decision Tree	0.6142	0.0192	0.6400	0.6270	0.0560	54
Rules	November Unique Page 56	Normalized To Self	Balanced Cascade	Decision Tree	0.6701	0.0188	0.5993	0.6337	0.0582	59
No Rules	December - Unique Page 86	Normalized To Self	Balanced Cascade	Decision Tree	0.6471	0.0073	0.6419	0.6445	0.0382	25
Rules	December - Unique Page 86	Normalized To Self	Balanced Cascade	Decision Tree	0.7206	0.0075	0.6133	0.6648	0.0435	32
No Rules	January - Unique Page 99	Normalized To Average User	Easy Ensemble	Decision Tree	0.6203	0.0372	0.6404	0.6303	0.0792	113
Rules	January - Unique Page 99	Normalized To Average User	Easy Ensemble	Decision Tree	0.6582	0.0369	0.6160	0.6368	0.0823	122
No Rules	April - Unique Page 5	Normalized To Average User	Balanced Cascade	Decision Tree	0.7164	0.0203	0.7011	0.7087	0.0837	165

Rules	April - Unique Page 5	Norm-a-lized To Average User	Balanced Cascade	Decision Tree	0.7711	0.0191	0.6576	0.7121	0.0830	162
No Rules	June - Unique Page 25	Nor-ma-lized To Average User	Easy Ensemble	Decision Tree	0.6220	0.0219	0.6491	0.6354	0.0630	78
Rules	June - Unique Page 25	Nor-ma-lized To Average User	Easy Ensem-ble	Decision Tree	0.6829	0.0213	0.6032	0.6418	0.0649	83
No Rules	Sep-tem-ber - Unique Page 45	Nor-ma-lized To Average User	Balanced Cascade	Decision Tree	0.7188	0.0079	0.6822	0.7002	0.0507	47
Rules	Sep-tem-ber - Unique Page 45	Nor-ma-lized To Average User	Balanced Cascade	Decision Tree	0.7500	0.0078	0.6677	0.7076	0.0522	50
No Rules	No-vem-ber - Unique Page 55	Nor-ma-lized To Average User	Balanced Cascade	Decision Tree	0.6310	0.0263	0.6311	0.6310	0.0671	78
Rules	No-vem-ber - Unique Page 55	Nor-ma-lized To Average User	Balanced Cascade	Decision Tree	0.6716	0.0255	0.5951	0.6322	0.0671	79
No Rules	De-cem-ber - Unique Page 75	Nor-ma-lized To Average User	Balanced Cascade	Decision Tree	0.9099	0.0586	0.7611	0.8322	0.1951	1338
Rules	De-cem-	Nor-ma-	Balanced Cascade	Decision Tree	0.9099	0.0586	0.7610	0.8321	0.1950	1337

	ber - Unique Page 75	lized To Average User								
No Rules	January - Unique Page 106	Normalized By Unique page	Easy Ensemble	Naïve Bayes	0.6667	0.0064	0.6606	0.6636	0.0396	29
Rules	January - Unique Page 106	Normalized By Unique page	Easy Ensemble	Naïve Bayes	0.7333	0.0065	0.6296	0.6795	0.0430	34
No Rules	April - Unique Page 6	Normalized By Unique page	Balanced Cascade	Decision Tree	0.6965	0.0212	0.7222	0.7092	0.0858	173
Rules	April - Unique Page 6	Normalized By Unique page	Balanced Cascade	Decision Tree	0.7015	0.0217	0.7268	0.7140	0.0882	183
No Rules	June - Unique Page 32	Normalized By Unique page	Balanced Cascade	Decision Tree	0.7250	0.0044	0.6563	0.6898	0.0366	26
Rules	June - Unique Page 32	Normalized By Unique page	Balanced Cascade	Decision Tree	0.6500	0.0043	0.6880	0.6687	0.0332	21
No Rules	September - Unique Page 44	Normalized By Unique page	Balanced Cascade	Decision Tree	0.7471	0.0128	0.7240	0.7355	0.0723	96
Rules	September - Unique	Normalized By	Balanced Cascade	Decision Tree	0.7471	0.0134	0.7372	0.7422	0.0754	104

	e Page 44	Unique page								
No Rules	No-vem-ber - Unique Page 55	Nor-ma-lized By Unique page	Balanced Cascade	Decision Tree	0.6458	0.0100	0.6405	0.6432	0.0444	34
Rules	No-vem-ber - Unique Page 55	Nor-ma-lized By Unique page	Balanced Cascade	Decision Tree	0.6875	0.0094	0.5929	0.6385	0.0425	31
No Rules	De-cem-ber - Unique Page 76	Nor-ma-lized By Unique page	Easy Ensem-ble	Decision Tree	0.6101	0.0252	0.6307	0.6203	0.0614	68
Rules	De-cem-ber - Unique Page 76	Nor-ma-lized By Unique page	Easy Ensem-ble	Decision Tree	0.5812	0.0255	0.6514	0.6153	0.0600	65

Table 5.3. Classifier performance with and without prescribed, hand-crafted rules.

We performed the Wilcoxon signed rank test on the 24 pairwise results to generate the Wilcoxon test statistic and associated standard score. The null hypothesis was that the rule-based method performed no better, or worse than the non-rule classifier and the alternative hypothesis was that the rule-base method performs better than the non-rule classifier. Performing this test yields a Wilcoxon test statistic of 80 and a standard score of -2.0, indicating that our results are significant at the 97.5% confidence level and that we can reject the null hypothesis that there is the classifiers with hand-crafted rules perform no better than classifiers without rules and accept the alternative that the hand-crafted rules outperform the classifier without rules.

5.4.3 Discussion

The results demonstrate that on the data sets tested, the rule-augmented classifiers outperformed those without rules. From this we conclude that **applying domain knowledge can increase our identification of rare action users**, but would not recommend a system based on creating rules by hand as a practical system.

Hand crafting rules is slow, experimentally questionable as it is easily subject to manipulation during testing as we can try out rule set after rule set until we get one that works. Furthermore, while the rule sets based classifiers outperform the classifiers without rules in this experiment, the difference between performance on any given dataset is not always exceptionally impressive. For example on dataset Page In Order June the difference in performance was only .0003. This is likely is possibly due to the fact that the rule-sets are not derived by examining each context a dataset is derived from, but rather is fed a pair of rule-sets bolted together by examining another set using the same normalization technique. Hence, this bolted on rule set is a reflection of different application context than users find themselves in when confronting the tasks before them and so they may not accurately reflect how prone users are to rare action in the applied context.

In fact, if we consider result differences less than .005 to be ties, the results are no longer significant at the 97.5% level, but instead the 95% level. Furthermore, if we consider difference less than .01 to be ties, are results are not significant by any reasonable confidence measure. This demonstrates just **how tenuously we can claim that hand-crafted rule sets are a general means of boosting performance**. We do argue, though, that **the results have shown that applying domain knowledge can be beneficial, and would further claim that if more time were spent crafting the rules to**

suit them to each particular survey page, the results would be more robust.

Note though that deriving a rule-set based on each concept (i.e. context) would be exceptionally slow, especially if one hoped to exploit all of the attributes available. This problem is more pronounced in regard to behavioral attributes, such as number of question responses, than it is for demographic traits such as age. While age's effect on rare action may be somewhat static, and time could be saved by applying the same age rules everywhere, the behavioral measures will certainly shift in regards to what kind of point is relatively high, or relatively low depending on context. Thus to fully exploit rule-set usage, rules based on behavioral attributes require tuning. In a real-time system this is unrealistic to do by hand. **This motivates the development of rule-extraction via automated processes.**

Exploration of beneficial rules is another source of motivation. Exploring rules by hand would be time consuming, and is a process which can easily be automated. We can also open up this exploration to search not a restricted set of ranges for numeric attributes, but instead a more open ended sit, as such a system is more robust against concept shifts that can develop within the system. As was pointed out in Chapter 2, there is evidence in the domain literature that domain expertise can actually invert itself due to context – i.e. older users struggle more with complex tasks, younger users are less motivated. This further motivates exploration against a set of hand-crafted limits on rule exploration, or the more extreme example of hand-crafted, static rule sets.

Furthermore, experimentally, automated rule-extraction can be shown to be generalizable, whereas that question of generalizability is doubtful for the study presented in this section for the reasons outlined above, namely the problem of manipulation and

derivation of rule-boundaries. **Automated rule-extraction can overcome this deficiency as we can extract rules from one set of data, and test the rule-set's performance on unseen testing data**. This is an important point if we want to prove that the secondary rule-technique does in fact work and can become a general part of a working user model.

In conclusion, we argue that this study has demonstrated that applying domain knowledge via rule-sets can be used to improve performance, but that a question yet to be answered is whether this domain knowledge can actually be extracted and applied in a generalizable, repeatable manner. To answer this question we engaged in the study presented in Section 5.5, in which rules are automatically extracted from training data, and then tested on unseen test data.

5.5 Automated rule-set performance

With this study we wanted to demonstrate that not only can domain knowledge be successfully applied to improving our rare-action user modeling problem, but that the knowledge can be extracted in **more** data-driven manner and to show that the process of extracting rule sets and applying them to classifiers is generalizable.

To summarize we had the following objectives:

Objective 3*:* Demonstrate that extracting rule-sets through a mixture of domain knowledge and data exploration succeeds in improving classifier performance. We argue that this is more feasible and realistic than the technique illustrated by Objective 2.

Objective 4*:* Demonstrate that automatic extraction of rule sets provides our user models with a repeatable, generalizable method of applying domain heuristics to the improvement of user classification as rare-actors.

In order to do this, we used the same 24 sets of data described in Section 5.4 and performed pairwise experiments in each data set comparing the performance of classifiers equipped with automatically extracted rule-sets to classifiers without such rule-sets. We then compared the results using the Wilcoxon-signed rank test and find that the rule-set equipped classifiers outperform the non-rule set classifiers at the 99.9% significance level.

5.5.1 Setup

Recall that in Section 5.3, we had previously studied classifiers without rules on the 24 datasets used in this study, and found the results (and associated resampling-classifier combination) with the highest g-mean for each of the normalization-month of survey pairs. And that these classifiers were evaluated using 10-fold cross validation. Since we performed cross-validation, we have 10 sets of data for each of the 24 top g-mean classifiers that represent how that classifier performs on unseen data. Since our rule-sets intend to adjust classifications made on unseen data, these 10 sets also represent potential training set and test-set sources for an experiment of the automated rule extraction and generation process.

In fact, using these 10 test-result tests we are able to devise a 10-fold cross validation strategy for testing automated rule extraction, by extracting and testing 10 different sets of rules on each of the 24 data-sets. This process was done by:

1. Applying the rule-extraction technique on nine of the ten original-classifier test results
2. Testing the extracted rules on the 10^{th} set of classifier results, to see how well the rule-sets overturn these classifications.
3. Repeat this process 10 times, using a distinct test set of classifier results each time.

Note that the classifier results tell us the original classification, and also provide us with the true value of the data. Our rule sets only examine the classifiers classification and the corresponding data instance (i.e., the associated user data) in determining whether to override the original classification. We can then see if the decision was correct by examining the true value.

In this way, we can see how well our classifier performs in terms of MCC, g-mean, and other statistics on each dataset and compare the general performance of rule-set equipped classifiers with of the original classifiers via the Wilcoxon-signed rank test.

5.5.2 Kinds of rules explored

In Chapter 3 we mentioned that in automated retrieval of rules we search a large field of prescribed boundaries for numeric attribute based rules, and explore prescribed sets of nominal-attribute values for boundaries associated with nominal attributes. This is precisely the approach we took in this study. We also limited the comparison operators we studied to the following:

- Value In Set for nominal attributes
- Value Not In Set for nominal attributes
- Less than for numeric attributes

- Greater than for numeric attributes
- Less than value 1 or greater than value 2 for numeric attributes

Thus, in the study of automated rule extraction and application presented in this Section we are not yet fully exploiting, nor exploring all the possible rules that can be generated for numeric attributes.

Also, recall that each rule-set can be equipped with a threshold that instructs the rule-set how many votes are needed to overturn the classifier's judgment. Rather than set the threshold in stone, we tested each possible threshold when testing extracted rule-sets on the unseen classification data. This accumulates more information for future work and allows us to see what impact varying thresholds has on rule-set performance. Each threshold was varied from the point at which no classifications would be overturned to the point where all classifications would be overturned. In comparing rule-set performance to classifiers without rule-sets, we excluded the examination of rule-set performance using thresholds that caused the rules to go unused. This way, each rule-set performance had to make use of the extracted rules.

The generated True Negative and True Positive rule sets contain at most one rule per attribute, though rules can be merged via the amalgamation process as described in Chapter 3. All attributes were explored as possible rule sources. A typical True Negative Rule Set is composed of 10-20 rules, a typical True Positive Rule Set tends to have the same number. On the whole, numeric attributes dominate the nominal attributes in terms of presence in a rule set. In most rule-sets, all but one to two of the rules are based on the numeric attributes – though age is an attribute commonly seen across rule-sets.

The relatively large size of the rule sets (often composing 25-50% of the attributes) and strong representation of numeric attributes gives support to the practice of using automated data exploration in conjunction with domain knowledge in finding rules. Note that constructing reliable rule-sets of this size and attribute-coverage by hand would not be practical due to the time required, and that the fact we were able to find rule-sets of this size demonstrates that there is in any dataset a large amount of potential rules possible to exploit. We claim this because for one, as the results will show, these rule-sets perform very well in boosting performance, and two the methodology that we use to find rules is performance-based: rules only make it into the final rule-set if they perform well on the training data.

5.5.3 Results

To demonstrate how the rule-sets perform vs. the classifiers with no rule-sets, we present the recall (on the rare-action cases), precision (on the rare-action cases), true-negative rate, g-mean, Matthew's correlation coefficient (MCC) and its associated Chi-squared statistic. For the reasons argued in Chapter 2 and in the introduction of this Chapter, we focus on performance in terms of MCC, recall, and g-mean to assess the results as these statistics are robust against class imbalance.

The results appear in Table 5.4 below which summarizes the results of using resampling and ML classifiers in identifying users as rare-action or non-rare action users in the Gallup panel, as well as the use of these techniques in conjunction with rule-sets extracted from a combination of domain knowledge and data search. The data sets used are the same presented in Section 5.1 and Section 5.2 and the key is the same for this table as table 5.1.

Rules or No Rules	Data Set	Norm. Tech-nique	Resam-pling	Classi-fier	Rec.	Prec.	True Neg. Rate	G-mean	MCC	χ^2
No Rules	January - Page 2	Page In Order	Easy Ensemble	Decision Tree	0.636	0.047	0.640	0.638	0.093	155
Rules	January - Page 2	Page In Order	Easy Ensemble	Decision Tree	0.661	0.048	0.631	0.646	0.098	171
No Rules	June - Page 10	Page In Order	Easy Ensemble	Decision Tree	0.640	0.021	0.628	0.634	0.061	71
Rules	June - Page 10	Page In Order	Easy Ensemble	Decision Tree	0.686	0.022	0.626	0.655	0.071	97
No Rules	April - Page 4	Page In Order	Bal-anced Cascade	Decision Tree	0.656	0.018	0.648	0.652	0.063	78
Rules	April - Page 4	Page In Order	Bal-anced Cascade	Decision Tree	0.692	0.019	0.647	0.669	0.070	97
No Rules	September - Page 13	Page In Order	Bal-anced Cascade	Naïve Bayes	0.814	0.008	0.753	0.783	0.064	74
Rules	September - Page 13	Page In Order	Bal-anced Cascade	Naïve Bayes	0.860	0.008	0.756	0.806	0.070	88
No Rules	November - Page 5	Page In Order	Easy Ensemble	Decision Tree	0.612	0.021	0.646	0.629	0.058	59
Rules	November - Page 5	Page In Order	Easy Ensemble	Decision Tree	0.655	0.022	0.639	0.647	0.067	76
No Rules	December - Page 9	Page In Order	Bal-anced Cascade	Decision Tree	0.684	0.012	0.679	0.681	0.058	59
Rules	December - Page 9	Page In Order	Bal-anced Cascade	Decision Tree	0.735	0.013	0.685	0.710	0.067	79
No Rules	January Unique Page 106	Nor-ma-lized To Self	Easy Ensemble	Decision Tree	0.741	0.027	0.680	0.710	0.097	275
Rules	January Unique Page 106	Nor-ma-lized To Self	Easy Ensemble	Decision Tree	0.762	0.027	0.678	0.719	0.101	299
No Rules	April - Unique Page 5	Nor-ma-lized To Self	Easy Ensemble	Decision Tree	0.721	0.020	0.701	0.711	0.085	169
Rules	April - Unique Page 5	Nor-ma-lized To Self	Easy Ensemble	Decision Tree	0.741	0.021	0.698	0.719	0.088	181

No Rules	June - Unique Page 30	Normalized To Self	Balanced Cascade	Decision Tree	0.642	0.015	0.654	0.648	0.056	31
Rules	June - Unique Page 30	Normalized To Self	Balanced Cascade	Decision Tree	0.716	0.018	0.663	0.689	0.073	51
No Rules	September - Unique Page 45	Normalized To Self	Easy Ensemble	Naïve Bayes	0.688	0.009	0.722	0.704	0.054	53
Rules	September - Unique Page 45	Normalized To Self	Easy Ensemble	Naïve Bayes	0.797	0.009	0.676	0.734	0.059	65
No Rules	November Unique Page 56	Normalized To Self	Balanced Cascade	Decision Tree	0.614	0.019	0.640	0.627	0.056	54
Rules	November Unique Page 56	Normalized To Self	Balanced Cascade	Decision Tree	0.665	0.020	0.626	0.645	0.063	70
No Rules	December - Unique Page 86	Normalized To Self	Balanced Cascade	Decision Tree	0.647	0.007	0.642	0.644	0.038	25
Rules	December - Unique Page 86	Normalized To Self	Balanced Cascade	Decision Tree	0.750	0.008	0.640	0.693	0.051	45
No Rules	January - Unique Page 99	Normalized To Average User	Easy Ensemble	Decision Tree	0.620	0.037	0.640	0.630	0.079	113
Rules	January - Unique Page 99	Normalized To Average User	Easy Ensemble	Decision Tree	0.641	0.038	0.640	0.640	0.085	131
No Rules	April - Unique Page 5	Normalized To Average User	Balanced Cascade	Decision Tree	0.716	0.020	0.701	0.709	0.084	165
Rules	April - Unique Page 5	Normalized To Average User	Balanced Cascade	Decision Tree	0.721	0.021	0.704	0.713	0.086	172
No Rules	June - Unique Page 25	Normalized To Average User	Easy Ensemble	Decision Tree	0.622	0.022	0.649	0.635	0.063	78

Rules	June - Unique Page 25	Normalized To Average User	Easy Ensemble	Decision Tree	0.675	0.023	0.634	0.654	0.071	99
No Rules	September - Unique Page 45	Normalized To Average User	Balanced Cascade	Decision Tree	0.719	0.008	0.682	0.700	0.051	47
Rules	September - Unique Page 45	Normalized To Average User	Balanced Cascade	Decision Tree	0.781	0.008	0.672	0.724	0.057	59
No Rules	November - Unique Page 55	Normalized To Average User	Balanced Cascade	Decision Tree	0.631	0.026	0.631	0.631	0.067	78
Rules	November - Unique Page 55	Normalized To Average User	Balanced Cascade	Decision Tree	0.642	0.027	0.629	0.636	0.069	84
No Rules	December - Unique Page 75	Normalized To Average User	Balanced Cascade	Decision Tree	0.910	0.059	0.761	0.832	0.195	1338
Rules	December - Unique Page 75	Normalized To Average User	Balanced Cascade	Decision Tree	0.912	0.059	0.761	0.833	0.196	1348
No Rules	January - Unique Page 106	Normalized By Unique page	Easy Ensemble	Naïve Bayes	0.667	0.006	0.661	0.664	0.040	29
Rules	January - Unique Page 106	Normalized By Unique page	Easy Ensemble	Naïve Bayes	0.833	0.009	0.695	0.761	0.066	78
No Rules	April - Unique Page 6	Normalized By Unique page	Balanced Cascade	Decision Tree	0.697	0.021	0.722	0.709	0.086	173
Rules	April - Unique Page 6	Normalized By Unique page	Balanced Cascade	Decision Tree	0.721	0.022	0.719	0.720	0.090	189
No Rules	June - Unique Page 32	Normalized By	Balanced Cascade	Decision Tree	0.725	0.004	0.656	0.690	0.037	26

		Unique page								
Rules	June - Unique Page 32	Nor-ma-lized By Unique page	Bal-anced Cascade	Decision Tree	0.825	0.005	0.673	0.745	0.048	45
No Rules	September - Unique Page 44	Nor-ma-lized By Unique page	Bal-anced Cascade	Decision Tree	0.747	0.013	0.724	0.735	0.072	96
Rules	September - Unique Page 44	Nor-ma-lized By Unique page	Bal-anced Cascade	Decision Tree	0.782	0.014	0.734	0.758	0.080	118
No Rules	November - Unique Page 55	Nor-ma-lized By Unique page	Bal-anced Cascade	Decision Tree	0.646	0.010	0.641	0.643	0.044	34
Rules	November - Unique Page 55	Nor-ma-lized By Unique page	Bal-anced Cascade	Decision Tree	0.688	0.011	0.640	0.663	0.051	44
No Rules	December - Unique Page 76	Nor-ma-lized By Unique page	Easy Ensemb le	Decision Tree	0.610	0.025	0.631	0.620	0.061	68
Rules	December - Unique Page 76	Nor-ma-lized By Unique page	Easy Ensemb le	Decision Tree	0.628	0.026	0.635	0.632	0.067	81

Table 5.4. Classifier performance with automatically extracted rule sets and without rule sets

By observing the MCC and its associated Chi-squared statistic we can see that **the rule-set equipped classifiers perform significantly better than random guessing**. We also see that g-mean and recall are boosted when using the rule-sets compared to not using the rule-sets. In fact, performing the Wilcoxon signed rank test on the results (in regards to g-mean) yields a Wilcoxon test statistic of 0 and a standard score of -4.29 (i.e. the rule equipped classifiers outperformed the non-rule classifiers on every pair). This

means that we can reject the null hypothesis that the rule-sets perform no better than the non-rule set classifiers at the 99.9% confidence level; we can thus accept the alternative hypothesis that **the rule-set equipped classification outperforms the non-rule set classification**. Also, note that the differences in performance in this experiment between rule-set equipped classifiers and non-rule set equipped classifiers are much higher than what was found for the prescribed rules. In fact, if we consider the absolute performance differences less than .01 to be ties within the Wilcoxon sign-ranked test, we still achieve a standard score -3.99 and a Wilcoxon score or 10.5; indicating significance at the 99.9% confidence level.

5.5.4 Discussion

The results of this study provide very strong evidence that rule-set equipped classifiers outperform non-rule set equipped classifiers on the data sets tested. We claim that this shows the first part of Objective 3 was accomplished, and that extracting rules via a mixture of domain knowledge and data exploration improves classifier performance. Furthermore, since the rule-sets were tested on unseen data, we argue that this implies the rules extracted from training data are proving to be generalizable on the unseen data, thus addressing Objective 4. Consequently, we argue that this gives weight to the claim that such a system can be reused in practice as this methodology could be reapplied to other datasets, where we could use a set of training data in order to arrive at rules that could be employed on unseen data.

This further implies that the automated rule set generation is much more feasible and practical than the technique presented in Section 5.4. Recall that we pointed out that hand-crafted rule-generation is slow, unlikely to be generalizable due to context

constraints, and that experiments using hand-crafted rules are questionable due to the ability of the researcher to spend an indefinite amount of time searching out an ideal rule-set, time we note they would not have in practice. The automated extraction, on the other hand, is capable of implicitly taking context into consideration by finding the rule sets based on empirical evidence, and is further able to do so quickly and consistently (that is running the same extraction algorithm nine times will yield the same nine rule-sets, whereas hand-crafted rules do not have this guarantee). From this we can see that the automated process is more generalizable, experimentally verifiable, and practical than its handcrafted and prescribed cousin. Furthermore, the machine can utilize the data to an extent far greater than a prescribed system can, as the automated algorithm can search the rule space in a fraction of the time it takes for an expert to write a single rule-set. Hence, another advantage of this technique is superior exploitation of the information available to the user model – an advantage that is further evidenced by the superior performance of the automated rule sets against the prescribed rule sets shown in Sections 5.4 and 5.5.

We also argue that the data-driven rule-extraction is still domain-driven as it relies on searching nominal attributes in prescribed manner based on logical relationships between values. Similarly for numeric attributes, the comparison operators used to search the data are designed in such a way that the values grouped together for overturning classifier output are consistent with each other in terms of representing one or two sets of extrema, or in future work, into groups representing specific regions of behavioral action. Because of this, we claim that the rule-extraction process is domain-driven, as we logically associate values in a manner consistent with domain knowledge. For example, consider a numeric attribute measuring response rate. The current rule extraction scheme

will associate slower or faster users into a common group, thus reflecting the domain knowledge that proceeding through a survey at a faster or slower rate possesses implications regarding user motivation, attention, and question comprehension. In this way, the rules search out boundaries for concepts reasonable from a domain-literature perspective. They do not seek out **odd** relationships that have **no basis on domain knowledge** but would simply reflect probable noise in the data such as: people with 2, 9, and 3 kids should be considered rare-actors.

Also, note that the rules extracted did not fully exploit all the possible rules that can be generated, as we limited the creation of numeric-attribute based rules to three operators, and didn't include operators that examine a range of data, such as $x > 10$ and $x < 12$. Instead, we focused on rule-sets that examine extreme attribute values with rules like $x < 2$; $x > 20$; or $x < 2$ or $x > 20$. Hence, we expect including more operators into the search would boost performance further, as are methodology sifts through candidate rules to find the most robust one. This motivates future work.

Open questions remain after this study, however. One important question is how much training data is needed until the rule-sets become viable. This is an important consideration if we want to apply the rule-set equipped classifiers to as many users as possible. It is in fact an important consideration for any future predictive scheme within our models, whether we stick with the rule, classifier combination, or not. Tests on more datasets would also be valuable in showing that the rule-sets outperform the non-rule set classifiers in a variety of situations.

Work will also need to be down in arriving at a threshold prior to applying the rule-sets to unseen data. Whereas this study allows the exploration of the threshold, in

real-time system we would ideally already have an idea about what kind of threshold will be needed. We can begin this work by examining the thresholds used in these results and seeing if some type of general governs what kind of threshold to employ. Furthermore, are rule-combining currently relies on simple voting. This might not be ideal either; and so more work will need to be done in exploring the best way to combine rules.

Despite these considerations, we do argue that this study **demonstrates that automated extraction of rules based on domain knowledge produces generalizable sets of rules that can be used to improve classifier performance**. Thus, we argue that this system, or a similar one, could be used to improve rare-action predictions made by ML classifiers.

5.5.5 A Brief Comparison to Markov Chains

This study compares the performance of the two-tiered "rule-based" technique presented in this thesis, to that of a Markov Chain Opponent in regards to classifying survey sessions as rare-action cases or non-rare action cases.

To our knowledge, there are no studies regarding the prediction of survey error in the literature. It was thus a challenge to find a base-line to compare the two-tiered technique against. That being said, techniques involving Markov Chains and Hidden Markov Models have been applied to cousin areas of sequence or process classification, such as predicting user emotion in Spoken Dialogue Systems (Englebrech et al (2009); or altering service in Automated Ticketing Systems (Englebrech et al (2009)).

For this reason, a Markov Chain Opponent classifier was chosen as a basis for comparison. The goal of this study was to investigate the hypothesis that the two-tiered filtering technique would perform no better than this alternative classifier.

Our opponent was comprised of two competing chains – one modeling the transition probabilities between survey states for breakoff users in Gallup, and one modeling the transition probabilities between survey states for non-breakoff users in Gallup.

To build the models, a sample set of surveys were modeled independently as a series of state transitions, or Markov Chains. Each state transition was then counted, and from these counts the transition probabilities were computed.

We then had two representative Markov Chains or transition matrices: (1) one for breakoff or "rare-action" surveys and (2) one for non-rare action (non-breakoff) surveys. These representative chains or matrices held the transition probabilities between each state. We then classified test surveys by determining whether each test survey had a higher probability of being a rare-action survey or non-rare action survey.

These Markov Chains were trained using two of the user behavior traits found to be strongly correlated with rare action:

- The number of questions appearing on the next page of the survey
- The number of questions skipped on the current page of the survey.

To construct Markov Chains, we view the survey as a sequence. Each member of the sequence represents actions taken by the user on a given survey page; characterized by the number of questions skipped by the user on that page, and the number of questions appearing on the subsequent page.

The Markov Chains were constructed in an exponential fashion, labeling question counts as:

- ONE

- TWO
- FOUR
- EIGHT
- MORE

and labeling Question Skips as:

- NONE
- ONE

Experimentally, the Markov Chains were constructed from the same sub-sequence of survey data that the two-tiered rule-based filtering technique learned from and used to classify surveys as rare-action or non-rare action. That is, they both learned by starting at the beginning of the survey and collecting data until the same, certain page was reached. The page-stopping points matched those used in this thesis' original study, and reflect areas where large numbers of users quit the survey. These page are of interest because they represent areas of the surveys where an adept classifier could have the most a positive impact on the performance of the system.

The specific pages used here were (all page numbers represent how many pages have been viewed unless it is indicated that the page is associated with a specific unique page):

- Page 4 of the April survey
- Page 10 of the June survey
- Page 13 of the September survey
- Unique page with ID 55 of the November survey
- Page 9 of the December survey.

The experiment was implemented by constructing transition matrices for breakoff and non-breakoff surveys from 90% of the gathered surveys, and then testing the effectiveness of the Markov Chain opponent technique on the remaining 10% of the surveys.

The results of this experiment are as follows, in Table 5.5.5: the recall, precision, true negative rate, and g-mean were used to compare the classifiers as these are common ways of measuring the performance of classifiers in the face of the class imbalance problem (He and Garcia, 2009). The recall and true negative rate give us an idea of how well the classifier performs on rare-action and non-rare action cases, respectively. The g-mean (the geometric mean of the recall and true negative rate) gives us a combined measure of performance on the rare-action and non-rare action cases. Finally, the precision illustrates, in this case, the fact that both classifiers tend to do a rather poor job making a stark distinction between the rare-action and non-rare action cases.

Classification Strategy	Recall	Precision	True Negative Rate	G-mean ($\sqrt{}$ (Recall*TN Rate))
April – Page 4 with Rules	.69	.018	.63	.66
April – Markov Chain	.038	.0088	.94	.19

June - Page 10 Rules	.64	.021	.62	.63
June – Markov Chain	.45	.039	.78	.59
September Page 13 Rules	.81	.0078	.75	.78
September Markov Chain	.75	.035	.64	.69
November - Page 55	.69	.011	.64	.66
November Markov Chain	.91	.019	.047	.21
December – Page 9	.74	.013	.69	.71
December Markov Chain	.70	.042	.69	.70

Table 5.5 – Rule Filtering Technique Performance and Markov Chain Performance

As can be seen the rule-based technique outperforms the Markov Chain Opponent. In fact, performing the Wilcoxon Signed Rank Test on this experimental data leads to a one-tailed test statistic of 15. Fifteen is the minimum test statistic for a sample size of five for rejecting the null hypothesis that the rule-based method is no better or worse at classifying behavior than the Markov Chain method at the 95% confidence

level; thus we can state that this study provides evidence that the rule-based method is superior to the Markov Chain Opponent technique.

Both the rule-based and Markov Chain Opponent techniques made use of the same subsequence of a given survey. These subsequences reflect the view a classifier could realistically have over a survey while it is being taken by a user. One advantage that the rule-based technique had over the Markov Chain Opponent technique is the use of domain-based heuristics that can leverage data in ways that the probability-based Markov Chain Opponent cannot.

Consider that the Markov Chain Opponent is restricted in its inference mechanism to counting the number of times rare-action and non-rare action surveys transition between various states. Similarly, the learning mechanisms used earlier in this thesis have restricted inference mechanisms.

We showed, though, that these inference mechanisms could be augmented with domain knowledge (i.e. rule-sets) in such a way that improves classifier performance. Here we again see evidence that domain knowledge can improve survey classification beyond levels achieved by data-based classification mechanisms.

While the Markov Chain Opponent might be quite good at classifying processes (see Appendix E for data showing the performance of the Markov Chain Opponent when viewing the entire survey), it seems to fall short when an incomplete picture of the process is presented to it. This makes sense with what has been observed about breakoff surveys elsewhere in this thesis – namely that breakoff happens early, and thus that the breakoff processes are marked by a much shorter process, and much more relevantly to the Markov Chain Opponent – fewer transitions between certain kinds of states.

Specifically, as a survey proceeds the Markov Chain constructed will certainly have more opportunities to hit states parameterized by certain question counts, as each page of the survey will be seen rather than just a subset. Due to this, the transition probabilities between users going "deep" into a survey versus those quitting early will definitely be quite different. Given now that breakoff users tend to depart fast, it is reasonable to infer that differences in this difference in the process will be picked up by a Markov Chain Opponent.

The study above though, does not have access to the entire process, but rather to a snapshot of it as a learner might face in real time. For this reason, certain vital information about the process is absent to the Markov Chain.

Thus information absence affects the Markov Chain in at least two segments of data:

1. Process data that the Markov Chain is built to handle, such as state transitions; and
2. Data that the Markov Chain is not built to handle, such as domain knowledge about surveys and breakoff.

These shortcomings conspire to handicap the Markov Chain versus the two-tiered rule-equipped filtering technique which is able to leverage domain knowledge, and does not rely on process modeling necessarily.

Thus, what this study has shown is that expanding the types of information available to a learner its abilities can augment its performance. In particular, we have seen that there is evidence that domain knowledge can improve classifier performance. This is especially crucial in the survey domain as we are dealing with the difficult task of trying to infer human behavior and mental engagement via quite limited observations; namely various metrics regarding survey participation or survey characteristics.

5.6 Identifying rare-actions in ATUS

The goal of this section is to further show the generalizability of the rule-set methodology by demonstrating that it boosts the detection of rare-action users engaging in ATUS. Data sets were derived for ATUS using the three normalization techniques described in Chapter 3. We then investigated the use of the resampling techniques combined with the same three ML classifiers used in the past experiments in detection of rare user actions, against the same techniques combined with the automated rule extraction and application techniques.

Another objective is to show that rare-action user modeling in ATUS is possible by demonstrating that user session data and user-demographic data can be successfully applied to classifying users as rare-actors or not rare-actors. We will show this by illustrating that the classifiers outperform random guessing significantly. We will also discuss the limitations of the current ATUS work, and give ideas about how it can be improved in the discussion section.

In summary we had the following objectives:

Objective 5: Show that the extracting rule-sets methodology can boost detection of rare-actions in multiple settings, by performing experiments with rule equipped classifiers and non-rule equipped classifiers in ATUS.

Objective 6: Demonstrate that rare-action centered user-modeling in ATUS is feasible, by proving that user-session data and demographic information can detect rare-actors at a rate significantly better than random.

In order to address this objective we used the same approach as in Section 5.5; performing a set of pairwise tests across multiple data sets and then performing the

Wilcoxon Signed Rank Test to compare rule against non-rule performance. We find that the rule-set classifiers outperform the non-rule classifiers at the 99.9% confidence level. As in Section 5.5, the rule-sets were derived by extracting them from training data which was independent of the data the rule-set classifier combinations were tested on. To accomplish this we used the same 10-fold cross validation technique as described in Section 5.5.

5.6.1 Setup

As just stated, the setup for this experiment was identical to that of Section 5.5 only this time engaging with ATUS data. Here, however, we did not restrict our attention to only the best performing classifiers for comparison, as there was not the same extreme amount of data sets due to the fact that we did not split the ATUS data up by page, as there is no analogous concept. Instead, the data is taken from the entire interview process.

This complicates interpretation of results as it is not clear yet whether this system is robust enough to be used in real-time; we can use these results however, to give more weight to the use of rule-sets and to show that rare-actions are detectable and predictable at a higher rate when using the two-tiered rule-set scheme. Since we are using holistic data, we exempted any attributes from the attribute-sets that would be a direct tip-off that a rare-action occurred.

In order to execute the experiment, the non-rule set classifiers were first built and tested with 10-fold cross validation. As in Section 5.5, these results could then serve as the training and testing sets for rule-extraction and rule-equipped classifier performance in a second 10-fold cross validation test. In the end, we thus had results from 10-fold cross validation performed on the non-rule set classifiers and the rule-set classifiers.

For the rule-sets, we again explored thresholds rather than restrict them as no methodology has yet been constructed to pre-determine the ideal thresholds, though it could be a matter as simple of investigating the training data for an idea of an ideal threshold. In the following two sections we first present the results and then discuss their implications.

5.6.2 Results

The results appear below in Table 5.5. The key for the table is the same as that of previous sections. Note that the MCC and Chi-squared statistic indicate that our results were significantly better than random guessing using the rule-set technique in each dataset (though not always by the non-rule technique), further evidencing that rare-user actions are detectable by models. Note too that the rule-equipped classifiers consistently outperform the non-rule equipped classifiers.

Rules or No Rules	Norm. Tech-nique	Resampling Technique	Classifier	Rec.	Prec.	True Neg. Rate	G-mean	MCC	χ^2
No Rules	Raw Count	Oversampling	Naïve Bayes	0.52	0.09	0.76	0.63	0.14	241
Rules	Raw Count	Oversampling	Naïve Bays	0.59	0.09	0.70	0.64	0.13	228
No Rules	Raw Count	Oversampling	Decision Tree	0.07	0.10	0.97	0.26	0.05	28
Rules	Raw Count	Oversampling	Decision Tree	0.58	0.06	0.57	0.57	0.06	46
No Rules	Raw Count	Oversampling	Multilayer Perceptron	0.06	0.09	0.97	0.05	0.04	19
Rules	Raw Count	Oversampling	Multilayer Perceptron	0.63	0.06	0.55	0.59	0.07	42
No Rules	Raw Count	Undersampling	Naïve Bayes	0.68	0.08	0.63	0.65	0.13	231
Rules	Raw Count	Undersampling	Naïve Bayes	0.71	0.08	0.63	0.67	0.15	279
No Rules	Raw Count	Undersampling	Decision Tree	0.64	0.07	0.62	0.63	0.11	157
Rules	Raw Count	Undersampling	Decision Tree	0.68	0.08	0.63	0.65	0.13	222
No Rules	Raw Count	Undersampling	Multilayer Perceptron	0.63	0.08	0.66	0.42	0.13	217

Rules	Raw Count	Undersampling	Multilayer Perceptron	0.66	0.08	0.66	0.66	0.14	255
No Rules	Raw Count	Ordinary Training	Naïve Bayes	0.49	0.09	0.78	0.62	0.13	224
Rules	Raw Count	Ordinary Training	Naïve Bays	0.56	0.10	0.75	0.65	0.14	272
No Rules	Raw Count	Ordinary Training	Decision Tree	0.00	0.10	1.00	0.04	0.01	1
Rules	Raw Count	Ordinary Training	Decision Tree	0.54	0.05	0.55	0.55	0.04	20
No Rules	Raw Count	Ordinary Training	Multilayer Perceptron	0.01	0.22	1.00	0.01	0.03	13
Rules	Raw Count	Ordinary Training	Multilayer Perceptron	0.58	0.06	0.54	0.56	0.05	15
No Rules	Raw Count	Easy Ensemble	Naïve Bayes	0.72	0.09	0.65	0.68	0.16	333
Rules	Raw Count	Easy Ensemble	Naïve Bayes	0.72	0.09	0.66	0.69	0.17	368
No Rules	Raw Count	Easy Ensemble	Decision Tree	0.79	0.10	0.64	0.71	0.19	461
Rules	Raw Count	Easy Ensemble	Decision Tree	0.79	0.10	0.65	0.72	0.19	491
No Rules	Raw Count	Balanced Cascade	Naïve Bayes	0.73	0.09	0.66	0.69	0.17	372
Rules	Raw Count	Balanced Cascade	Naïve Bayes	0.73	0.10	0.67	0.70	0.18	404
No Rules	Raw Count	Balanced Cascade	Decision Tree	0.79	0.10	0.64	0.71	0.19	454
Rules	Raw Count	Balanced Cascade	Decision Tree	0.79	0.10	0.65	0.72	0.19	480
No Rules	To Average	Oversampling	Naïve Bayes	0.56	0.09	0.74	0.64	0.14	250
Rules	To Average	Oversampling	Naïve Bayes	0.61	0.09	0.71	0.66	0.14	272
No Rules	To Average	Oversampling	Decision Tree	0.11	0.14	0.97	0.33	0.09	98
Rules	To Average	Oversampling	Decision Tree	0.52	0.06	0.61	0.56	0.06	40
No Rules	To Average	Undersampling	Naïve Bayes	0.68	0.08	0.62	0.65	0.13	212
Rules	To Average	Undersampling	Naïve Bayes	0.70	0.08	0.62	0.66	0.14	255
No Rules	To Average	Undersampling	Decision Tree	0.66	0.08	0.63	0.64	0.12	199
Rules	To Average	Undersampling	Decision Tree	0.70	0.08	0.62	0.66	0.14	244
No Rules	To Average	Ordinary Training	Naïve Bayes	0.50	0.09	0.76	0.62	0.13	205
Rules	To Average	Ordinary Training	Naïve Bays	0.58	0.09	0.72	0.64	0.14	239
No Rules	To Average	Ordinary Training	Decision Tree	0.00	0.00	1.00	0.00	0.00	0
Rules	To Average	Ordinary Training	Decision Tree	0.61	0.05	0.43	0.51	0.02	3

No Rules	To Average	Easy Ensemble	Naïve Bayes	0.72	0.09	0.65	0.68	0.16	331
Rules	To Average	Easy Ensemble	Naïve Bayes	0.73	0.09	0.66	0.69	0.17	370
No Rules	To Average	Easy Ensemble	Decision Tree	0.79	0.10	0.64	0.71	0.19	456
Rules	To Average	Easy Ensemble	Decision Tree	0.80	0.10	0.65	0.72	0.19	480
No Rules	To Average	Balanced Cascade	Naïve Bayes	0.72	0.09	0.66	0.69	0.16	354
Rules	To Average	Balanced Cascade	Naïve Bayes	0.73	0.09	0.66	0.70	0.17	384
No Rules	To Average	Balanced Cascade	Decision Tree	0.79	0.09	0.64	0.71	0.18	446
Rules	To Average	Balanced Cascade	Decision Tree	0.79	0.10	0.65	0.72	0.19	474
No Rules	To Entry Count	Oversampling	Naïve Bayes	0.75	0.06	0.46	0.59	0.09	102
Rules	To Entry Count	Oversampling	Naïve Bayes	0.67	0.07	0.58	0.62	0.10	144
No Rules	To Entry Count	Oversampling	Decision Tree	0.08	0.10	0.97	0.28	0.05	33
Rules	To Entry Count	Oversampling	Decision Tree	0.57	0.06	0.59	0.58	0.07	59
No Rules	To Entry Count	Oversampling	Multilayer Perceptron	0.09	0.12	0.97	0.09	0.07	61
Rules	To Entry Count	Oversampling	Multilayer Perceptron	0.61	0.06	0.56	0.58	0.07	64
No Rules	To Entry Count	Undersampling	Naïve Bayes	0.77	0.06	0.42	0.57	0.08	89
Rules	To Entry Count	Undersampling	Naïve Bayes	0.65	0.07	0.58	0.61	0.10	119
No Rules	To Entry Count	Undersampling	Decision Tree	0.62	0.07	0.59	0.61	0.09	106
Rules	To Entry Count	Undersampling	Decision Tree	0.65	0.08	0.63	0.64	0.12	193
No Rules	To Entry Count	Undersampling	Multilayer Perceptron	0.66	0.07	0.56	0.37	0.09	110
Rules	To Entry Count	Undersampling	Multilayer Perceptron	0.66	0.07	0.59	0.63	0.11	150
No Rules	To Entry Count	Ordinary Training	Naïve Bayes	0.76	0.06	0.45	0.58	0.08	94

Rules	To Entry Count	Ordinary Training	Naïve Bayes	0.70	0.07	0.55	0.62	0.11	149
No Rules	To Entry Count	Ordinary Training	Decision Tree	0.01	0.23	1.00	0.07	0.03	10
Rules	To Entry Count	Ordinary Training	Decision Tree	0.55	0.05	0.54	0.55	0.04	19
No Rules	To Entry Count	Ordinary Training	Multilayer Perceptron	0.01	0.14	1.00	0.01	0.03	9
Rules	To Entry Count	Ordinary Training	Multilayer Perceptron	0.50	0.06	0.61	0.55	0.05	19
No Rules	To Entry Count	Easy Ensemble	Naïve Bayes	0.73	0.08	0.60	0.66	0.14	261
Rules	To Entry Count	Easy Ensemble	Naïve Bayes	0.73	0.09	0.64	0.68	0.16	319
No Rules	To Entry Count	Balanced Cascade	Naïve Bayes	0.72	0.08	0.61	0.66	0.14	258
Rules	To Entry Count	Balanced Cascade	Naïve Bayes	0.71	0.09	0.64	0.67	0.15	299
No Rules	To Entry Count	Balanced Cascade	Decision Tree	0.72	0.09	0.66	0.69	0.17	363
Rules	To Entry Count	Balanced Cascade	Decision Tree	0.72	0.10	0.67	0.70	0.17	390

Table 5.6. Results of ATUS user classification, rule set technique and non-rule set technique

Performing the Wilcoxon signed rank test (and counting differences less than .01 as ties) on the data yields a Wilcoxon test statistic of 22.5 and an associated standard score of -4.79 showing that our results are significant that the 99.9% confidence level and that the rule-set results were statistically superior to the non-rule set results.

5.6.3 Discussion

The results give further evidence that equipping classifiers with automated rule extraction techniques combining domain knowledge and data exploration delivers a generalizable,

repeatable way of boosting user classification as now we have two distinct problems that have had performance improved by the application of that methodology. For user modeling within ATUS, these results should be taken with a grain of salt, as even though the data sets were derived via session recreation and user demographics, they represent statistics derived from the entire session. A practical method of predicting user error in ATUS will likely have to be tweaked in order to base the classifiers in a sequential fashion – as either a series of classifiers based on the number of activities seen thus far, or as a sequential technique itself. As with Gallup, we had initially investigated the use of long short-term memory neural networks in predicting rare-user actions in ATUS, but the results were so poor we altered course. It might also be possible to use the classifiers developed here, particularly the normalized ones as their take on the data is based on the average action on an activity or entry, but we have left these investigations to future work.

Nevertheless, we claim that we have shown that data extracted from user-sessions and regarding users can be used to make predictions that are capable of classifying users as rare-actors as non-rare actors. From this perspective, we have engaged in a proof-of-concept study in this Section by demonstrating that the rare-action problem is addressable via data derived from a user-modeling process that can be subsequently used to derive more robust techniques.

The ATUS datasets have a large amount of attributes based on incidences within a user session – that total 93 attributes. Similarly, the rule-sets extracted for ATUS are quite large as well – typically containing 70 or more rules, with at most one rule per attribute. The high amount of rules shows that the data-search is capable of finding ways

of exploiting a large proportion of the numeric attributes in a way beneficial to user classification, giving more credence to its introduction in place of hand-crafted sets. Such large rule sets though, raise questions of if they are currently being combined in the most ideal fashion.

As mentioned before, we currently use a simple combination scheme utilizing voting. Exploring alternative combination strategies could yield better results, such as a form of weighted voting. Also, as with the study in Section 5.5, we limited the rule operations to search the extreme ends of the numeric attributes for rules rather than hunting for ranges. Opening up the data exploration further, could also yield better rules. Given that data exploration used in Section 5.5 outperforms prescribed rules in Section 5.4, work to engage in more exploration is further motivated.

5.7 Feature Extraction - Context

Another goal of this thesis was to extract features of the environment associated with rare actions. To do this, we constructed Pearson correlation coefficients between the amount of rare action occurring on a page, and the presence of the feature. There are two ways to consider what page a rare-act occurred on in the Gallup panel, as the rare-action in question is users quitting surveys. We can consider the rare act page to either the last page we have data for (or the last page the user submitted back to the Gallup server in other words), or the last page the user likely saw, i.e. the expected page after the last submitted page. Since it is not clear which page should be considered more associated with the rare-action we examine both forms and present correlations for both.

The objective of this study was the following:

Objective 7: Determine if any environmental features are associated with the

presence of rare-action in the Gallup panel.

5.7.1 Setup

The Pearson correlation coefficient is a simple way to determine whether to variables are dependent on one another. Since we are interested in determining if there is any relationship between survey features, or user context, and the presence of rare user actions, computing the correlation coefficient between the amount of rare action on a page and the presence of certain page features is a simple way we can begin this exploration.

We present correlations between the rare-act of survey quitting and the following survey-page features:

1. The page number of the survey – this gives us an idea about whether rare actions increase, or decrease as the survey proceeds.
2. The number of questions on the survey page – pages with more questions are considered more difficult and are thought to affect factors associated with rare action such as motivation, or cognitive ability.
3. The number of words on the survey page – this gives us an idea about the complexity of questions on the page.
4. The Flesch reading ease associated with the text on the survey page – this also gives an idea about question complexity.
5. The number of grid-type questions on the survey page – recall that grid type questions are when one base question is provided and the user must answer this question repeatedly with some part of the question changing in each sub-question in the grid. For example, here is a possible grid question: Describe

your reaction to the following hamburgers: Big Mac, Whopper, Whataburger, …, etc. Grid questions are known to be associated with error.

6. The number of sub-questions belonging to grids on the survey page. This is similar to the number of grid-questions, but we count the total number of sub-questions instead. This gives us a better idea of the size of the grids and allows us to investigate if large grids, or very many small grids, are associated with rare action.
7. The number of times the topic changes on a survey page. Topic changes are thought to be unpleasant to survey takers, and thus could affect the underlying causes of rare-actions in surveys.

5.7.2 Results

Table 5.6 below shows the Pearson correlation coefficients between page features and the incidence of rare user action on the page on which the rare action occurred. The rare action is considered as occurring on the last page the user likely saw, or in other words, the page after the last page we have user data on.

Table 5.7 below shows the Pearson correlation coefficients between page features and the incidence of rare user action on the page on which the rare action occurred. The rare action is considered as occurring on the last page the user interacted completely with, i.e. the last page for which we have interaction data for. Table 5.8 shows the amount of rare action occurring on a given page (in terms of the last page the user interacted with).

	Pearson Correlation Coefficient						
Page Feature	April Survey	June Survey	September Survey	November Survey	December Survey	January Survey	All Surveys
Page Number	.11	.67	-.37	-.19	-.081	-.46	-.15
Word Count	.30	.39	.20	.34	.62	.63	.30
Flesch Reading Ease	.55	-.23	-.054	.027	.10	.16	-.11

Number of Grid Questions	.25	.34	.38	.11	.069	.25	.22
Total number of questions in Grids	.12	.43	.86	.79	.19	.71	.34
Number of Topic Changes	.13	-.37	0	.23	.53	-.071	-.045
Number of Questions	.15	.45	.84	.26	.33	.67	.40

Table 5.7. Pearson correlation between page features and rare-action count, where rare-action pages are based on the last page likely seen by the users.

	Pearson Correlation Coefficient						
Page Feature	April Survey	June Survey	September Survey	November Survey	December Survey	January Survey	All Surveys
Page Number	-.28	.46	-.53	-.37	-.37	-.56	-.28
Word Count	-.016	.47	-.22	.012	-.083	-.053	-.005
Flesch Reading Ease	-.32	.017	.69	.18	.17	.27	.091
Number of Grid Questions	.005	.81	.073	-.093	-.14	.53	.21
Total number of questions in Grids	.043	.79	.15	-.044	-.052	.28	.21
Number of Topic Changes	-.19	.36	.031	.10	.25	-.075	-.12
Number of Questions	.029	.81	.079	-.012	-.12	.073	.17

Table 5.8. The Pearson correlation coefficient between survey page features and the amount of rare-actions occurring on the page. Here, the rare-action page is considered the last page the user submitted.

April		June		September		November		December		January	
Page Name	Rare Ac-tion Count	Page Name	Rare Ac-tion Count	Page Name	Rare Ac-tion Count	Page Name	Rare Ac-tion Count	Page Name	Rare Ac-tion Count	Page Nam e	Rare Ac-tion Count
q1	71	q24	82	sa1	95	q2	61	q7	56	q11	131
q3	47	q23	45	q5	62	q3	37	q1	53	q4	121
q10_1	32	q22	35	q6	26	q29	27	p4	38	sa1	87
q17	27	q20	34	q18	22	sa1	24	q2	36	q1	26
p4	24	p4	34	login	20	login	20	sa1	23	q5	25
q5	21	q1	20	q4	20	p4	20	login	19	q23	25
q5_1	18	q29	20	q13	19	q37	20	q3	14	login	24
q10	8	q13	13	q21	18	q1	19	q8	14	q13	18
q13	6	q16	9	q12	14	q16	18	q14	8	q19	10
q24	6	q15	7	p4	14	q14	14	q19	8	q10	8
q6	2	q7	7	q1	12	q35	13	q18	6	q31	8
q18	2	q18	7	q26	5	q15	12	q13	5	q24	6
q25	1	q24_1	4	q7	4	q19	11	q32	5	q29_ 1	6
login	0	q32_1	5	q10	4	q28	8	q5_t	3	q32_ 5	4
sa1	0	q24_3	2	q9	3	q9	4	q25	3	q33_ 1	3
q23	0	login	0	q34	3	q27	2	q28	3	q22	2
		sa1	0	q35	1	q34	2	q31	3	q26	2
				q32	0	q13	3	q6	2	q28	2
						q20	3	q15	2	q32	1
						q31	2	q6_t	0	q32_ 3	1
						q30	1	q22	0	q32_ 4	0
						q33	0	q27	0		

Table 5.9. Rare action counts per page by Month.

5.7.3 Discussion

As can be seen, we present the data both for individual surveys and for the correlations across all surveys. Interestingly, the two statistics most consistently correlated with rare actions are (1) the number of grid questions, and (2) the total number of questions in grids. This trend is most pronounced when considering the rare-action page to be the last

page likely seen, but still exists to a lesser degree when considering the last page submitted to be the rare action page.

Note also, that the general trend is for most rare-action (i.e., breakoff) to occur early in the survey, hence the negative correlation with page number. Interestingly, the June panel is the exception to this trend as the opposite effect is found. Consistent positive, and often moderate to strong correlations, are found between word count and question count and rare actions, when taking the last page seen perspective. Returning to our experimental objective, we claim the results show that survey features can indeed be found that are associated with rare-action.

These observations match domain knowledge, and demonstrate that it may be possible to develop systems that engage users more aggressively on survey pages deemed to be rare action inducing. If we can validate these trends across future data sets, we can coax our design, predictive measures, and interfaces to address pages entertaining high amounts of traits correlated with rare action to be more acceptable; either by offering assistance of some form, or by having a system alert methodologists of possible concerns.

To further back up the assertion that certain pages can be found to be troublesome and potentially addressed, observe Table 5.3, where it is apparent that certain pages in each survey dominate the incidence of breakoff. This observation has several more interesting implications.

First, on top of a system that utilizes correlations, or some other method (perhaps more machine learning or data mining techniques) to predict *a priori* what pages will induce rare action, it appears that it is reasonable to expect that we can observe, during a survey's lifetime, which pages breakoff is occurring on. This alone provides us with an

opportunity to adjust our system as we accumulate these observations. This is actually an important consideration regarding future design as well, because responsiveness to such observations requires not only data tracking but a software design by which interfaces can be easily changed or adapted as observational data accrues. Hence, if these ideas are followed in future work, the system design must consider them at a very basic level.

Another implication of certain pages dominating the amount of rare action, is that if other prediction techniques are developed for the rare-action user modeling problem, (other than the series of ML classifiers we present), they should be designed to pay heed to either survey page characteristics, or the raw probability (via the observational technique just outlined) of rare actions when making decisions, as it can be seen that the incidence rate of rare-actions is more prominent in certain contexts. For example, if in the future a hidden Markov model (HMM) is found to be a superior predictor of rare-action, these results imply that the HMM might be able to increase its performance by either being augmented with a secondary module that utilizes context-based rare-action likelihood (either predicted or observed) as a way of adjusting the HMM's predictions, or by somehow incorporating the likelihoods into the model itself.

In conclusion, we were able to show that survey features can in fact be correlated to incidence of rare-action, with respect to Objective 7. More importantly, we argue that this has very interesting implications for future work, both in regards to our current methodology, as we can inform the complete system to pay more attention to our current predictive mechanism on pages with very high likelihood of rare action, and in regards to future possible models.

5.8 Feature Extraction – Behavior based

We can also extract features of user behavior and the user demographics related to rare-action in order to detect interesting and useful trends in the data. We do this in a similar manner to Section 5.7, developing the Pearson correlation coefficients between user characteristics or behavior and the presence of rare-action. We do this for both ATUS and the Gallup panel.

These investigations had the following objective:

Objective 8: Determine if rare-actions are associated with certain trends in user-behavior and certain user-demographics.

5.8.1 Setup and Results

To perform this investigation we accumulated data regarding the Gallup panels in their final state of completion for each user – thus acquiring a bulk picture of their actions. Analogous information already existed for ATUS via the study described in Section 5.6. From these data sets we then derived the Pearson correlation coefficients between each attribute and the presence of that survey's rare action. We present the findings in Tables 5.9, 5.10, and 5.11, below, omitting results deemed too weak (i.e., correlations with absolute value below .05) in order to focus our discussion on the findings. In each of the tables below, a negative correlation indicates that the higher the attribute value, the more likely a user is to commit a rare-action, whereas positive correlations indicate a decreased likelihood as the attribute's value increases.

January Gallup Panel Pearson Correlation Coefficients (R)		**April Gallup Panel Pearson Correlation Coefficients (R)**		**June Gallup Panel Pearson Correlation Coefficients (R)**	
Attribute	**R**	**Attribute**	**R**	**Attribute**	**R**
Percent of Total Time Spent Scrolling	-0.05	Age	0.07	Percent of Grid Questions Straightlined	-0.05
Average Question Response Time	-0.05	Number of Skips	0.08	Number of Scrolls	0.05

Percent of Grid Questions Straightlined	0.06	Number of Scrolls	0.09	Age	0.05
Age	0.07	Percent of Grid Questions Straightlined	0.11	Number of Skips	0.05
Submission Time	0.09	Number of Pages Submitted	0.15	Average Question Response Time	-.06
Question Responses Per Page	0.10	Number of Straightlines	0.15	Number of Straightlines	0.11
Question Response Time	0.11	Number of Question Responses Per Page	0.29	Number of Questions Shown Per Page	-0.15
Number Straightlined	0.11	Number of Question Responses	0.42	Number of Question Responses Per Page	0.20
Number of Scrolls	0.12	Number of Questions Shown Per Page	0.45	Number of Item Nonresponses	-0.30
Number of Item Non-response	-0.12	Number of Questions Shown	0.58	Percent Answered	0.31
Total Time	0.13	Number of Pages Downloaded	0.65	Number of Pages Submitted	0.33
Number of Backups	0.16			Item Non-Response Per Page	-0.40
Percent Answered	0.19			Number of Question Responses	0.46
Number of Skips per Page	0.19			Number of Questions Shown	0.61
Number of Questions Shown Per Page	-0.23			Number of Pages Downloaded	0.62
Number of Skips	0.34				
Number of Item-non response per page	-0.42				
Number of page submission actions	0.49				
Number of Question Responses	0.64				
Number of Questions Shown	0.67				
Number of Pages Downloaded	0.83				

Table 5.10 Behavioral and demographic attribute correlations to rare action in the Gallup panel for months: January, April, and June.

September Gallup Panel Pearson Correlation Coefficients (R)		December Gallup Panel Pearson Correlation Coefficients (R)		November Gallup Panel Pearson Correlation Coefficients (R)	
Attribute	**R**	**Attribute**	**R**	**Attribute**	**R**
Percent of Total Time Spent Answering Questions	0.05	Number of Scrolls	0.05	Number of Skips	0.05
Average Scroll Time	-0.05	Age	0.05	Percent of Total Time Spent Answering Questions	0.05
Total Page Submission Time	0.06	Average Question Response Time	-0.06	Age	0.06
Total Download Time	0.06	Item Non-Response Per Page	0.06	Number of Scrolls Per Page	0.06
Number of Skips	0.07	Average Page Submission Time	-0.07	Total Page Download Time	0.07
Age	0.08	Percent Answered	-0.07	Number of Question Responses Per Page	0.08
Percent of Grid Questions Straightlined	0.11	Percent of Total Time Spent Answering Questions	0.07	Number of Scrolls	0.14
Number of Scrolls	0.12	Number of Skips	0.09	Number of Question Responses	0.20
Number of Pages Submitted	0.15	Number of Item Nonresponses	0.10	Number of Questions Shown Per Page	0.27
Number of Straightlines	0.17	Number of Backward Navigations Per Page	0.11	Number of Questions Shown	0.40
Number of Question Responses Per Page	0.22	Number of Question Responses Per Page	0.14	Number of Pages Submitted	0.61
Number of Questions Shown Per Page	0.31	Percent of Grid Questions Straightlined	0.14	Number of Pages Downloaded	0.70
Number of Question Responses	0.62	Number of Straightlines	0.15		
Number of Pages Downloaded	0.79	Number of Backward Navigations	0.15		
Number of Questions Shown	0.84	Number of Question Responses	0.18		
		Number of Questions Shown Per Page	0.19		
		Number of Questions Shown	0.20		
		Number of Pages Submitted	0.24		
		Number of Pages Downloaded	0.26		

Table 5.11 Behavioral and demographic attribute correlations to rare action in the Gallup panel for months: September, November, and December.

Pearson Correlation Coefficients (R) in Normalized To Average User Data Set		Pearson Correlation Coefficients (R) in Normalized To Number of Activity Level Edits Data Set		Pearson Correlation Coefficients (R) in Raw Data Set	
Attribute	**R**	**Edits**	**R**	**Counts**	**R**
Time Type Changes	-0.05	Number of Activity Inserts	-0.05	Time Type Changes	-0.05
Number of Activities Occurring at Work	0.05	Number of Error Prompts	0.05	Number of Activities Occurring at Work	0.05
Number of Activities done Alone	-0.05	Number of Activities Occurring at Work	0.05	Number of Activities done Alone	-0.05
Number of Minute Duration Changes	-0.06	Number of Where Information Changes	0.06	Number of Minute Duration Changes	-0.06
Number of Activity Inserts	-0.06	End of days triggered	0.06	Number of Activity Inserts	-0.06
Number of Times Given as Duration	-0.06	Changes to with Whom Information	0.06	Number of Times Given as Duration	-0.06
Total Activities created During Interview	-0.06	Hours Worked Per Week	0.07	Total Activities created During Interview	-0.06
Hours Worked Per Week	0.07	Number of With Whoms Kept Private	-0.07	Hours Worked Per Week	0.07
Number of Activities	-0.07	Number of Activities	-0.07	Number of Activities	-0.07
Number of Activity Level Edits	-0.07	Number of Wheres Kept Private	-0.07	Number of Activity Level Edits	-0.07
Number of Activities Done at Home	-0.07	Number of Activity Level Edits	-0.07	Number of Activities Done at Home	-0.07
Number of Activity Type Edits	-0.08	Number of Activity Times Given as Durations	0.07	Number of Activity Type Edits	-0.08
Age	-0.08	Age	-0.08	Age	-0.08
Number of With Whoms Kept Private	-0.18			Number of With Whoms Kept Private	-0.18
Number of Wheres Kept Private	-0.19			Number of Wheres Kept Private	-0.19

Table 5.12. Correlations between behavioral and demographic attributes and rare-action in ATUS.

5.8.2 Discussion

As can be seen in the above Tables 5.9, 5.10, and 5.11 age was weakly correlated with rare action for both Gallup and ATUS. Also, we can see more evidence for the Gallup panel that the breakoff rare-action tends to happen early in the survey as the correlation between the number of full downloads (i.e., pages seen), number of full downloads (again, pages seen), and other indicators of distance into the survey are among the most highly correlated with rare action presence.

Even counts and amounts per page of potential measurement errors like skips and straightlines are correlated with not engaging in rare action. We suspect that this is due to the fact that as the user proceeds further into the survey these numbers have a natural tendency to pile up. For instance, a skip is not a flag of non-response, it is an indication that user answered a question out of order. Thus, the skip measurements in these data-sets do not equate necessarily with non-response as these datasets were derived from a picture of an entire survey session. Hence, we should actually expect users who complete surveys to have a higher number of incidences where they answer questions out of order (i.e. skips). Similarly for straightlines, users who complete the surveys encounter more grid questions and have more opportunities to straightline. Also, recall that straightlining is only a potential measurement error, and so encountering more straightlines from users who complete the entire survey should not be unexpected. In fact, both of these measures are further evidence that rare-actions (i.e., breakoffs) happen early.

The January and June Gallup panels offer the strongest indications that measurement error observation is correlated with rare-action giving weight to the idea that the theoretical overlap between the causes of measurement error and rare-actions can be exploited in models by detecting measurement error. The measurement errors

observed in these surveys to be correlated are the percent of questions seen which were answered, and the amount of item non-response (both bulk and per page). In the other surveys we did not observe much correlation between measurement error and the rare-action. This might again be due to the fact that breakoff seems happen early, in general. Interestingly, in the June survey rare-actions tended to happen late (as indicated by the Page Number correlation shown in Section 5.7) and it was also one of the surveys where measurement error was correlated to breakoff. Besides age, no demographic features appear strongly correlated to rare-action in the Gallup panel.

The fact that the rare-actions tend to happen early is further supported by simply observing how quickly users tend to breakoff in most surveys. From a time perspective, Figure 5.1 below shows that comparing rare-action users to all users, the rare-action (breakoff) users spend a much shorter amount of time in the survey than typical users. The plot shows what percentage of the user group has completed participating in the survey after a given amount of time. In other words, we can see what percentage of all rare-actors have broken off after a given amount of time in contrast with what percentage of all users have completed the survey. (Since rare-actors constitute 2-3% of all users viewing the all user time-data is not significantly different from viewing only non-rare actor data). Note too that for most surveys, some 70% of rare-action users have already broken off within only 5 minutes. The one exception is again June, but we see that for even June the rare-action users complete their interactions with the surveys more quickly than the typical user.

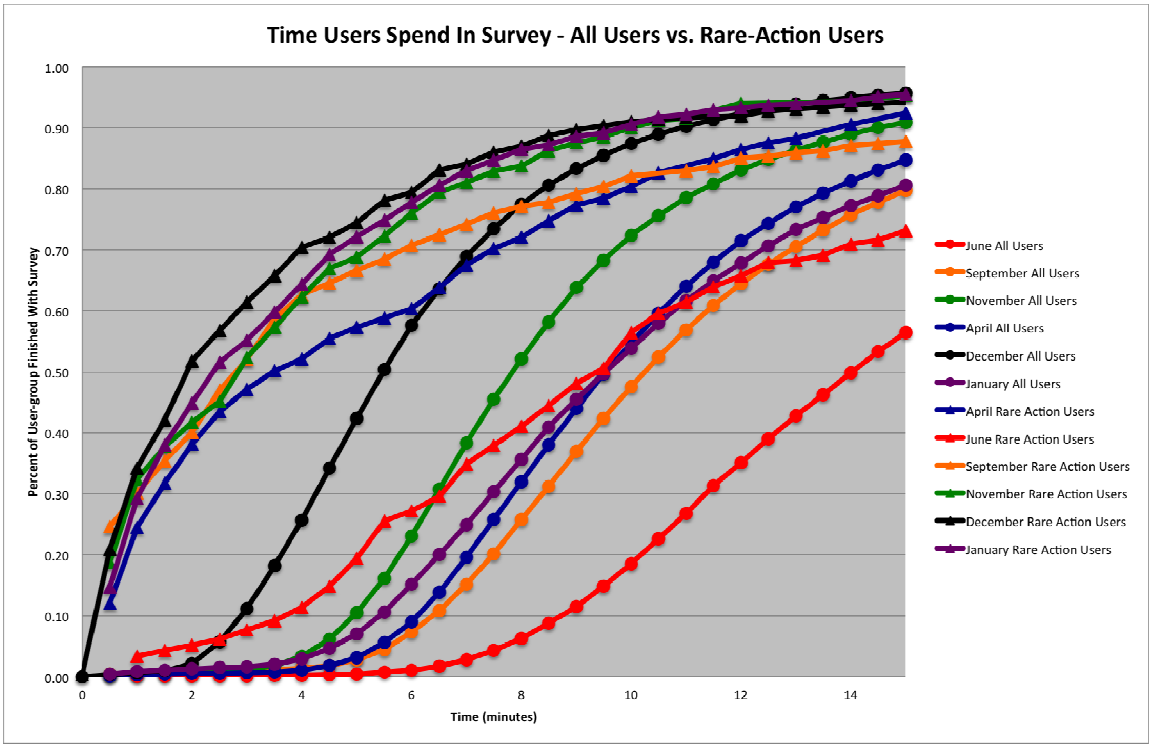

Figure 5.1. Percent of User-Group Having Completed Survey-Interaction Given Recorded Total Time Spent on Survey

The implication of the short interaction time for rare-actors and the fact the rare-action is breakoff, meaning they quit the survey before completing - is that we must identify, and engage rare-actors very quickly, in order to build an effective system that addresses the bulk of the rare-action error. Any future system will need to take this into account when building adaptations, when making use of predictions, and when designing alternative predictive mechanisms. In fact, even the current methodology might be confusing the constructed classifiers by having them attempt to identify *any* future rare-actor, instead of rare-actors we expect within a certain time or page window. We avoided this technique in this study as we feared doing so would increase the class imbalance problem. However, this might have come at the cost, in retrospect, of **damaging the ability of the ML classifiers to make sense of the data as rare-action users appear to**

quit the survey in a logarithmic fashion, implying that there may be substantial differences between rare-action users themselves. In other words, we may actually be dealing with **several sub-concepts of rare-action users, rather than a single group of rare-action users with common traits and behaviors**.

This has many interesting implications:

1. If it can be found that early rare-actors are quite different than late-term rare-actors, then **independent** predictive mechanisms could be constructed for each. Doing so could be as simple as keeping the current methodology intact (i.e., a series of classifiers equipped with secondary filtering techniques) but changing the way we label the data for training, and consequently learn how to identify users who will perform rare-actions within the next 60 seconds or within the next 2 pages, for example.
2. We observe in the plot, that as time goes on the curves for all-users and rare-action users approach one-another. This might imply that at some point in the survey, perhaps in terms of time or page-depth, rare-action uses may begin to resemble non-rare action users more and more. Hence **late-term separation of the two groups may be more difficult than early term**. This reemphasizes the notion that late-rare actors and early-rare actors may require separate predictive mechanisms.

Due to these observations, we will emphasize ideas for future work in Chapter 6, centered around the early rare-action problem.

On the ATUS side, age is modestly correlated to the presence of rare-action, verifying the domain knowledge that older users are more likely to forget what activities

they had engaged in. Most strongly correlated however, are the proportion of activities that are deemed too personal to include with whom or where indications. These activities have their presence noted by the value given by the "with whom" and "where" attributes – and are indicated by the value "kept private". Activities like these include sleeping and grooming.

The amount of editing that is done within an ATUS interview is also mildly correlated to rare-action presence, indicating that rare-actions (memory failures) tend to happen in interviews in which the interviewer is asked to re-edit or change information. This notion is further supported by the fact that edits of specific types – such as editing the activity type – are also correlated with rare-action presence. Performing many (or a large proportion) of activities at home is also correlated with rare-actions in ATUS.

In regards to Objective 8 we have shown that **some trends in user behaviors are indeed associated with rare-actions**. One interesting finding is that **the Gallup rare-action is strongly associated with occurring early in a survey**, and that the **ATUS rare-action is tied to the presence of editing and a higher proportion of personal and at home activities**. Findings like this can be used to equip user modeling systems with a better comprehension of when to take more aggressive action and encourage redesign that addresses issues within the proper time frame, page depth, or editing-level of the survey or interview. The other behavioral findings have similar implications – namely (1) **measurement can serve as a good flag for a user model to adjust its level of alertness in certain contexts (i.e., certain Surveys such as June and January)** and (2) **observing users engaging in a higher proportion of personal and at home activities are likely prone to rare action and may require more attention**.

5.9 Summary

A fair question is how the overall scheme used in these studies, involving classifiers both with and without domain based rules, suits the particular eccentricities of the domain we deal with, which we characterize by users possessing: a lack of self-direction, a limited ability to navigate content, and a restricted content set that all users must engage with.

The implications of these traits are that any user-modeling system is intrinsically limited in **1) the ways it can assist users**, and **2) the observations it can make regarding users.** For example, stating that users lack self-direction means that users do not choose which goals to pursue within the system, and instead are forced into pursuing a singular goal of survey completion. The consequence of this is that a modeling system can not render help by suggesting new goal-directions for users, nor can it model each user's individual interests and goals as the nature of the survey-system forces each user to pursue the same ends, and the same goal. Hence, there is no way for our model to benefit from goal-detection as most modeling systems can. Instead, **we can only attempt to view behaviors likely to be harmful to the one, shared goal all users - despite any individual traits - are forced to pursue.**

The implications of user's being forced to deal with a restricted set of content parallel the implications of users possessing a lack of self-direction in terms of goals. Our system cannot benefit by observing the content a user chooses to engage with, chooses to repress, or chooses to investigate, because every individual is forced into dealing with the same content. There is no analogous way in surveys to build user models like one might find in an online bookstore, which can be based on interests in specific content - such as horror novels or romance novels. Instead we must base our observations based on **how**

the users engage with content - and hence we develop our behavioral observations in terms of attributes like skipping questions, rushing through questions, and other entities that distinguish users **not** by **what** they engage with but by **how** they engage content. Furthermore, since we cannot alter the content of the surveys, there is no way for us to exploit perceived user interest; and instead we must focus only on behaviors that we infer to be indicative of negative user actions, i.e., the rare actions.

One can see that in regards to both self-direction and content restriction, we are impeded by a lack of choices available to users, and a lack of alternate routes. The same theme applies to navigation of surveys. In most systems, users can arrange the order they view content in any manner they see fit. In fact, there is no concept of a linear navigation path—the user model simply observes users exploring a complex web of linked content, for example. Returning to the online bookstore example, though the store hopes users will eventually make their way to the purchase page, there is no set way to land there. One user might haphazardly look at picture frames catalogues, followed by an almanac of potato guns, and concluding with a history book before finally making a purchase; whereas another user might jump straight to an item of interest and buy it. Users are wide open to take any path through the store they want; and in fact it is in the store's interest to let them explore as it allows users to view content of their choosing - thus providing information to models and letting users make their way to products they deem valuable. Surveys starkly contrast this; as there is a set, linear path all users must take. This is obvious to see in Gallup as users are all exposed to the same series of questions, but even ATUS can be characterized this way as every user must build the same time-diary spanning the same time span, with the same starting and ending points. A night owl can't

describe his day according to what happened to him after he got up at 9 am and until he went to bed at 2 - instead he must describe his day starting and ending at the same points an early morning shift worker describes theirs. They must both start their diary at 4 am and end it at 4 am. Because of this, our system cannot infer individual traits based on navigation, and consequently cannot make use of these non-existent observations. Instead we can only discern user differences based on **how** they steer themselves through the prescribed linear, course of navigation.

Due to these characteristics, our models focused on building datasets scented on user behaviors within common contexts (recall the normalization schemes) and in comparison to common measures. We also incorporated demographic traits as a way of distinguishing traits. These were deemed viable options in the face of barren data regarding explicit interests, preferences, and goals. Finally in ATUS, features of the surveys can be incorporated in *some* fashion by observing how entries are made into the survey and what kinds of data are derived. In regards to Gallup, the survey context is implicitly incorporated into the models since each dataset is based on user actions within a common context.

In summary, we were able to address each objective with some degree of success. However, we certainly possess reservations about the state and performance of the current techniques. Feature extraction enlightens these critiques further, providing valuable direction for future work. In the next Chapter, we will overview recommendations for next steps and studies.

In regards to the rare-action user modeling problem, we believe it has been demonstrated that both the resampling and rule-set techniques can detect rare-actions

with some degree of success and that the rule-sets derived from domain and data knowledge enhance this process. We further claim that the rule-set technique has proved generalizable as it works on unseen data and on diverse datasets. However, these techniques can no doubt be improved upon and so in Chapter 6 we will return the implications and ideas for future work briefly discussed in this Chapter.

Chapter 6

Conclusions & Future Work

In this chapter we summarize the findings of this thesis and then overview ideas for future work. Section 6.1 will review the conclusions and Section 6.2 will present the ideas for future work.

6.1 Conclusions

This thesis had several goals:

1. To demonstrate that user modeling can be applied to systems characterized by a **lack of self-direction**, **limited content navigation**, and **the presence of restricted content**.
2. To develop predictive mechanisms constructed with ML classifiers and domain-knowledge based rule-sets capable of predicting rare-actions users

take in such systems; and to show that the application of the rule-sets improves prediction performance.

3. To extract features of the surveys and users associated with the rare-actions we attempt to predict.

The results of Chapter 5 show that there user-modeling is feasible in systems characterized by the traits listed above in **1)**, as we were able to build classifiers for user-models able to group users into rare-actors and non rare-actors at a rate significantly better than random guessing. We were also successful in accomplishing goal **2)** as we showed that the application of domain based rule-sets improves the classification of users. We also were able to extract information associated with rare-actions, thus achieving goal **3)**.

Recall that we define lack of self-direction to mean that the user has limited control over their goals within the system. That is, in surveys, users have a singular goal of survey completion whereas most user-modeling applications exist within systems where the user self-directs towards goals of their own choosing. Furthermore, limited content navigation implies that users do not have a large degree of control over how they proceed through the system. Within surveys, users must complete a set, series of regimented tasks that they have no ability to alter. In contrast, typical user-modeling applications exist in systems where the users decide what to view, or do and when to view or do it. Similarly, content is restricted to the set of tasks associated with the survey, whereas in typical user modeling applications the user posses the power to alter the content, change the content they are viewing, and the system can even offer assistance by omitting content that is unpleasant to the individual. Since users must complete all parts

of a survey, the systems we deal with do not have these traits. Each of these characteristics distinguishes the task of building user models for surveys and similar systems from the traditional user modeling task.

Due to these unique features, the framework consisting of ML classifiers, resampling techniques, and rule-sets was designed to incorporate certain features that could differentiate users despite the forced similarities of each users interaction with the program. To do this, we strived to develop a framework gauging how users interacted with the limited, regimented environment as a means of distinguishing between user behaviors; and further designed the framework to measure these behaviors at the same contextual point for all users. In this way, we attempted to build predictive mechanisms that were attentive not to what content users prefer or tend towards (as in typical user modeling applications) but instead was attentive to differences in actions towards common, forced content, and forced pathways. We also utilized user demographics, implicitly considered the environment in Gallup by building a series of classifiers, and explicitly considered the survey state in ATUS.

To summarize we were able to show that **ML classifiers both with and without the secondary rule-sets can be applied to the user-modeling problem associated with surveys – namely one characterized by a lack of self-direction**, **limited content navigation**, and **the presence of restricted content**; and that the **application of domain knowledge via the rule-sets improves the predictive performance of these models**. We showed this by demonstrating the both techniques are statistically superior to random guessing, but that the application of rule-sets improves performance significantly at the 99.9% confidence level. We also showed that **the rule-set technique is generalizable as**

we successfully applied used rule-sets extracted from one set of data onto unseen data; and as we demonstrated that the technique improves performance in both the ATUS and Gallup domains.

In regards to goal **3)** we found that certain types of questions – namely grid questions – are associated with the prevalence of rare-actions occurring on a Gallup survey page. We also found that observing survey measurement errors is correlated to rare-action presence in some survey contexts, and that measures such as word count and question count are also associated with rare-action prevalence. These findings inform us that **user-models can benefit from being context-aware** in our domain and that **survey designers should carefully weigh decisions such as how many grid questions to display on a given page**. We can also enhance surveys, using this knowledge, by offering simple adaptations on pages likely to be difficult by doing things such as encouraging the user to finish by providing indications of their completeness on a page (thus encouraging them to finish) or by offering clarifying dialogue on questions with a high word count.

We also found that in Gallup the break off rare-action occurs early in the surveys lifetime – with some 70% of users breaking off within the first 5 minutes. This means that our **predictive mechanisms need to be especially effective early in the survey if we hope to assist most users**, and that future forms of assistance – such as clarifying dialogue or offering amusing questions in addition to the required ones – should be in place early in the survey as well. It also has implications for future work in regards to prediction which we will discuss in the next section.

In ATUS we found that certain activities with **certain types of activity where and with whom information is associated with error**. This implies that future models in ATUS should pay heed to these values, and also serves as evidence for survey methodologists to pursue in their understanding of causes of the memory gap rare action. We also found that **age and a higher amount of survey editing are associated with rare-action**, implying that models for ATUS should pay attention to these traits.

Also in regards to ATUS, while we did not yet construct a predictive mechanism that can likely be used in practice (as we utilized information extracted from entire surveys rather than up to a point in time), we did demonstrate that **data extracted from session recreation and known data at the time of the survey can be used to classify users as rare or non-rare actors and** hence demonstrated that **the user-modeling problem aimed at addressing rare-actions and keeping users heading towards their goals can work in ATUS as well as the Gallup panel**.

6.2 Future Work

In our opinion future work can center on several facets:

1. Exploring what kinds of adaptations to build into future survey versions.
2. Maturing predictive mechanisms to improve classification of users as rare or non-rare actors.
3. Designing survey software and data recording such that explicit feedback can be used to inform future models.

6.2.1 Exploring Adaptations

Future work exploring adaptations is a very wide open problem. We can use the work by Conrad et al (2007) as early guidance. Recall that Conrad et al (2007) offered users

clarifying dialogue with some success. Thus we can incorporate this design trait into the Gallup panel as well for long questions, or complicated, multi-part questions such as grid questions. This will also allow us to address the environments found to be associated with rare actions.

Recall that rare-actions in the Gallup panel are related to root causes of motivation, opportunity, and ability. Because of this, adaptations should center on addressing issues. In motivation for example, survey methodologists have found that encouraging users to keep going in a survey can positively effect behavior, and thus offering encouraging dialogue to users deemed likely to quit might be a way to address some breakoff users. Recall that opportunity can be related to things like technical problems. Thus one way we can address the opportunity problem is to observe issues like slow download time and take action when such observations are made – perhaps by offering the user the option of receiving the survey in less complex form. Another technical problem is the fact that the surveys are designed for desktop or laptop computers and not mobile devices – we can address this issue by adapting the survey's presentation when a mobile device is detected.

Ability issues relate back to difficult or unpleasant questions and things like mental fatigue. Again, clarifying dialogue can be used in such situations, as can things like a set of more diverting questions amended to the original survey. Perhaps mental fatigue could also be addressed by offering users the chance to see what the current results of the survey looks like once they complete the survey – this might encourage users to finish out of curiosity and the mental stimulation that would come from seeing how others think about the questions asked in the survey.

It is more difficult to suggest adaptations in ATUS as our computer-science group does not have the expertise to design memory queuing mechanisms. Hence, future work in this area will involve teaming with domain experts in memory in order to design beneficial behaviors and adaptations. We can envision simple adaptations in ATUS that address errors other than rare-action, such as missing information, that will simply function by calling timely attention to such omissions during an interview. Similar features currently exist in ATUS, but the omissions are not brought to the attention of the users until the entire time diary has been filled out; brining these things to light earlier would make the task of going back editing activities less burdensome. This will be especially important in future work as ATUS will be transformed from an interview to the same web-based version that Gallup employs (in other words there will be no interview, just a respondent filling out a time diary on a computer alone), and thus we will want to help users fill out their diaries as efficiently and easily as possible.

6.2.2 Maturing Predictive Mechanisms

The maturation of predictive mechanisms is another interesting area of future work. The current system, though proving the feasibility of user modeling, is not ideal. We achieve g-mean classification measures in the 65-72% range typically, and would like to improve this performance. We suspect that this performance ceiling may be developing due to the potential of sub-concept existence among rare-action users.

Remember that in Chapter 5 we mentioned that there are apparently two versions of rare-actors in Gallup – rare-actors who quit very early (the bulk of rare-actors) and those who quit later in the survey at times more consistent than typical users. This observation implies that there may be distinct types of rare-actors in Gallup and that the

current system is inherently handicapped by the fact that each classifier attempts to identify all rare-action users, rather than a distinct concept of a rare-action user.

To explore this potential dilemma, future work can be done that changes the way training data is labeled to a manner which only labels users who will engage in rare-actions within a certain time or page window from the current point in the survey as rare-actors. Since the current scheme in Gallup is a series of classifiers, doing this might allow each classifier to focus only on the most relevant sub-group of rare-action users. Doing this will have the consequence of increasing the class imbalance problem, and so it will be interesting to see if this positively or negatively effects the results. Also, if subgroups do exist it might imply that late-term rare-actors are more like non-rare actors than early-term rare-actors are, and thus the detection problem might become more difficult as time goes on and thus we might have to explore alternative means of detecting late-term rare-actors than used to identify early-term rare-actors, as different information might be relevant and as distinguishing might require more information.

Speaking of information, recall that a variety of normalization techniques were applied to the datasets. We suspect that future work could be done in building datasets from applying different normalization techniques to different attributes, depending on which technique is most appropriate for each attribute. This could lead to better information being fed into the classifiers and thus better performance. We suggest then that future work should involve determining which normalization to apply to each attribute.

Also recall that we suspect the rule-set performance can be improved by expanding the types of comparison operators performed during rule extraction. This is

also an area that future studies can make use of, and like the labeling shift, would be simple to implement.

Future work will also need to be done in ATUS in order to 1) test if the current modeling strategy works in real time (which we suspect it won't) and then 2) transform the current modeling strategy into series of classifiers analogous to those in Gallup. In order to do this, we suggest building a dataset, and thus a classifier, for each activity-level depth of an ATUS survey. That is, build one classifier and dataset for users currently 1 activity into the diary, another for users 2 activities in, and so on.

Of course the current work being based on a series of classifiers and datasets might not be ideal either. Thus future work should also be done that re-explores sequential methods (recall that we briefly mentioned these did not perform well in initial studies). While early work did not have much luck in using such techniques, there is now a much better picture of the problem than existed when those studies were conducted. For example, we have suspicions regarding sub-concepts of rare-actors, we know that certain environmental and behavioral traits are associated with rare-action, and we have seen that domain knowledge can improve classification performance. All these things can be leveraged in re-examining sequential and other alternative techniques.

Thus in regards to maturing the predictive mechanisms, work could be done that **first determines if altering the labeling scheme and rule-exploration improves performance, and then attempts to improve performance by building other types of classifiers using the knowledge that can be found in the current classifiers, rule-sets, and feature extraction work**.

6.2.3 Designing Survey Software and Data Recording

In future work we can also make use of explicit feedback. Specifically, we can view explicit feedback regarding the effectiveness of the adaptations we try out in order to learn which adaptations to accept and keep as part of the system, and which to get rid of. In other words, we can equip the software with **adaptive mechanisms aimed at affecting the rare-actions, observe how the user engages with the adaptive mechanism, observe if the rare-action is prevented, and thus use this series of events as explicit feedback regarding the utility of adaptations**. We can also **apply this explicit feedback in learning about the effectiveness of simpler adaptations**, such as ones aimed at showing users what questions they have skipped in the Gallup panel, or aimed at showing users what information they have omitted in ATUS.

6.2.4 Alternatives to Classification

Given the difficulties encountered in developing classifiers for categorizing users in the Gallup and ATUS surveys, an alternative approach that might bear more fruit would be to focus future work on the user-interface of computer based survey systems. Applying well-known usability techniques to the Gallup and ATUS surveys could lead to a study wherein the amount of rare-action in surveys designed with and without these usability considerations are compared.

Initially, such a study could be fairly simple and based on the existing literature describing features of suable software systems. Following this, the research team could devise its own mechanism of gathering "paradata" or observing users, and use data collected from this endeavor to refine the user interface further. Given the weak performance of the classifiers presented in this thesis, this area might be worthy of exploration before committing more work to the classification angle.

6.3 Summary

In conclusion, user modeling has been applied to an environment not typically associated with the user-modeling paradigm. The primary output of the models currently constructed are predictive mechanisms capable of detecting rare-user actions that are damaging to the user's interaction with the system, and the extraction of features of users, and the system context that are affiliated with such actions. The current models still have much room for improvement, but given the current findings we believe that such improvement is possible, and have thus laid out ideas for future work in this Chapter directed towards this goal.

References

American Time Use Survey (ATUS) Coding Rules 2008. Retrieved December 5, 2012, from http://www.bls.gov/tus/tu2008coderules.pdf.American Time Use Survey User's Guide, 2012. Retrieved October 15, 2012, from http://www.bls.gov/tus/atususersguide.pdf.

Armentano, M. G., & Amandi, A. A. (2012). Modeling sequences of user actions for statistical goal recognition. *User Modeling and User-Adapted Interaction*, *22*(3), 281-311.

Baldi, P., Brunak, S., Chauvin, Y., Andersen, C. A., & Nielsen, H. (2000). Assessing the accuracy of prediction algorithms for classification: an overview. *Bioinformatics*, *16*(5), 412-424.

Birukou, A., Blanzieri, E., & Giorgini, P. (2012). Implicit: a multi-agent recommendation system for web search. *Autonomous Agents and Multi-Agent Systems*, *24*(1), 141-174.

Biswas, Pradipta, and Peter Robinson. (2010) A brief survey on user modeling in HCI. *Proc. of the International Conference on Intelligent Human Computer Interaction (IHCI) 2010*.

Bonissone, P. P., Subbu, R., Eklund, N., & Kiehl, T. R. (2006). Evolutionary algorithms+ domain knowledge= real-world evolutionary computation. *Evolutionary Computation, IEEE Transactions on*, *10*(3), 256-280.

Bosnjak, M., & Tuten, T. L. (2001). Classifying response behaviors in web□based surveys. *Journal of Computer□Mediated Communication*, *6*(3), 0-0.Brusilovsky, P. (2001). Adaptive hypermedia. *User modeling and user-adapted interaction*, *11*(1-2), 87-110.

Chandola, V., Banerjee, A., & Kumar, V. (2009). Anomaly detection: A survey. *ACM Computing Surveys (CSUR)*, *41*(3), 15.

Chawla, N.V., Bowyer, K.W., Hall, L.O., and W. Philip Kegelmeyer. (2002). SMOTE: synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research 16*(1), 321-357.Chen, Z., Lin, F., Liu, H., Liu, Y., Ma, W. Y., & Wenyin, L. (2002). User intention modeling in web applications using data mining. *World Wide Web*, *5*(3), 181-191.

Conrad, F. G., Schober, M. F., & Coiner, T. (2007). Bringing features of human dialogue to web surveys. *Applied Cognitive Psychology*, *21*(2), 165-187. Couper, M. P. (2000). Review: Web surveys: A review of issues and approaches. *The Public Opinion Quarterly*, *64*(4), 464-494.

Cordier, A., Mascret, B., & Mille, A. (2010). Dynamic case based reasoning for contextual reuse of experience. In *Provenance-Awareness in Case-Based Reasoning Workshop. ICCBR,* (pp. 69-78).

Cortes, C., & Vapnik, V. (1995). Support-vector networks. *Machine learning*, *20*(3), 273-297.Curzon, P., & Blandford, A. (2001). Detecting multiple classes of user errors. In *Engineering for human-computer interaction* (pp. 57-71). Springer Berlin Heidelberg.Demšar, J. (2006). Statistical comparisons of classifiers over multiple data sets. *The Journal of Machine Learning Research*, *7*, 1-30.

Dillman, D. A., & Bowker, D. K. (2001). "The web questionnaire challenge to survey methodologists." Retrieved July 27, 2013 from http://mres.gmu.edu/pmwiki/uploads/Main/oss-book.pdf#page=59.

Engelbrech, K. P., Gödde, F., Hartard, F., Ketabdar, H., & Möller, S. (2009, September). Modeling user satisfaction with hidden markov model. In *Proceedings of the SIGDIAL 2009 Conference: The 10th Annual Meeting of the Special Interest Group on Discourse and Dialogue*. (pp. 170-177). Association for Computational Linguistics.

Fischer, G. (2001). User modeling in human–computer interaction. *User modeling and user-adapted interaction*, *11*(1-2), 65-86.

Frias-Martinez, E., Chen, S. Y., & Liu, X. (2006). Survey of data mining approaches to user modeling for adaptive hypermedia. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, *36*(6), 734-749.

Fricker, Scott. 2007. The Relationship between Response Propensity and DataQuality in the Current Population Survey and the American Time UseSurvey. Doctoral Dissertation, University of Maryland.

Fricker, S., Galesic, M., Tourangeau, R., & Yan, T. (2005). An experimental comparison of web and telephone surveys. *Public Opinion Quarterly*, *69*(3), 370-392.

Galesic, M., & Bosnjak, M. (2009). Effects of questionnaire length on participation and indicators of response quality in a web survey. *Public Opinion Quarterly*, *73*(2), 349-360.

Gutierrez, C., Wells, T., Rao, K., & Kurzynski, D. (2011). Catch Them When You Can: Speeders and Their Role in Online Data Quality. Paper presented at the Midwest Association of Public Opinion Research, Chicago, IL.

He, H., & Garcia, E. A. (2009). Learning from imbalanced data. *Knowledge and Data Engineering, IEEE Transactions on*, *21*(9), 1263-1284.

Heerwegh, D. (2003). Explaining response latencies and changing answers using client-side paradata from a web survey. *Social Science Computer Review*, *21*(3), 360-373.

Horvitz, E., Breese, J., Heckerman, D., Hovel, D., & Rommelse, K. (1998, July). The Lumiere project: Bayesian user modeling for inferring the goals and needs of software users. In *Proceedings of the Fourteenth conference on Uncertainty in artificial intelligence* (pp. 256-265). Morgan Kaufmann Publishers Inc..

Jameson, A. (2009). Adaptive interfaces and agents. *Human-Computer Interaction: Design Issues, Solutions, and Applications*, *105*.

Johnson, T. P., Fendrich, M., & Mackesy-Amiti, M. E. (2010). Computer Literacy and the Accuracy of Substance Use Reporting in an ACASI Survey. *Social Science Computer Review*, *28*(4), 515-523.

Kapoor, A., Burleson, W., & Picard, R. W. (2007). Automatic prediction of frustration. *International Journal of Human-Computer Studies*, *65*(8), 724-736.

Knäuper, B., Belli, R. F., Hill, D. H., & Herzog, A. R. (1997). Question difficulty and respondents' cognitive ability: The effect on data quality. *Journal Of Official Statistics, Stockholm*, *13*, 181-199.

Kobsa, A. (2001). Generic user modeling systems. *User modeling and user-adapted interaction*, *11*(1-2), 49-63.

Krosnick, J. A. (1991). Response strategies for coping with the cognitive demands of attitude measures in surveys. *Applied cognitive psychology*, *5*(3), 213-236.

Lakiotaki, K., Matsatsinis, N. F., & Tsoukiàs, A. (2011). Multicriteria user modeling in recommender systems. *Intelligent Systems, IEEE*, *26*(2), 64-76.

Lops, P., de Gemmis, M., & Semeraro, G. (2011). Content-based recommender systems: State of the art and trends. In *Recommender Systems Handbook* (pp. 73-105). Springer US.

Liu, X. Y., Wu, J., & Zhou, Z. H. (2009). Exploratory undersampling for class-imbalance learning. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, *39*(2), 539-550.

Malpani, A., Ravindran, B., & Murthy, H. (2011, March). Personalized Intelligent Tutoring System Using Reinforcement Learning. In *FLAIRS Conference*.

Manavoglu, E., Pavlov, D., & Giles, C. L. (2003, November). Probabilistic user behavior models. In *Data Mining, 2003. ICDM 2003. Third IEEE International Conference on* (pp. 203-210). IEEE.

Mitchell, Thomas M. (1997). *Machine Learning*. New York, NY, USA: McGraw-Hill, Inc..

Neapolitan, Richard E. (2004). *Learning Bayesian Networks*. Saddle River, NJ: Pearson Education Inc.

Nicholls, W., Baker, R., & Martin, J. (2012). The effect of new data collection technologies on survey data. *L. Lyberg et al*, 221-44.

Pardos, Z. A., Heffernan, N. T., Anderson, B., Heffernan, C. L., & Schools, W. P. (2010). Using fine-grained skill models to fit student performance with Bayesian networks. *Handbook of Educational Data Mining*, 417.

Pardos, Z. A., & Heffernan, N. T. (2010). Modeling individualization in a Bayesian networks implementation of knowledge tracing. In *User Modeling, Adaptation, and Personalization* (pp. 255-266). Springer Berlin Heidelberg.

Pennacchiotti, M., & Popescu, A. M. (2011, May). A Machine Learning Approach to Twitter User Classification. In *ICWSM*.

Peytchev, A. (2009). Survey breakoff. *Public Opinion Quarterly*, *73*(1), 74-97.

Phillips, L. A., T. Baghal, and R. Belli. Troubles with Time-Use: ExaminingPotential Indicators of Error in ATUS. (Draft at this time, not yet published).

Pohle, C. (2003, September). Integrating and Updating Domain Knowledge with Data Mining. In *VLDB PhD Workshop*.

Salem, M. B., & Stolfo, S. J. (2011, January). Modeling user search behavior for masquerade detection. In *Recent Advances in Intrusion Detection* (pp. 181-200). Springer Berlin Heidelberg.

Shani, G., & Gunawardana, A. (2011). Evaluating recommendation systems. In *Recommender systems handbook* (pp. 257-297). Springer US.

Sinha, A. P., & Zhao, H. (2011). Tuning expert systems for cost-sensitive decisions. *Advances in Artificial Intelligence*, *2011*, 3.

Virvou, M., Troussas, C., & Alepis, E. (2012, June). Machine learning for user modeling in a multilingual learning system. In *Information Society (i-Society), 2012 International Conference on* (pp. 292-297). IEEE.

Yan, T., & Tourangeau, R. (2008). Fast times and easy questions: the effects of age, experience and question complexity on web survey response times. *Applied Cognitive Psychology*, *22*(1), 51-68.

Appendices

There are four appendices. In Appendix A the Gallup database is presented. Appendices B and C describe the attributes used in the Gallup and ATUS datasets, respectively. Appendix D details session recreation in ATUS.

A. Gallup panel database

We store session, user, and survey data in a relational database for the Gallup panel, using MySQL. Figure A below presents the schema of the database. In total there are 29 tables, which are structured to encapsulate the logical relationships between data. Tables represent things like users in terms of demographics, questions found in surveys, pages found in surveys, surveys found in the panel's history, types of respondent actions, and any other entity we make use of in our models.

To communicate this idea, the figure groups tables into the concepts: respondent (user) information (i.e. demographics), survey information (i.e. the content and context the users engage with – like questions and pages), responses to survey questions, paradata information (that is the log-file entries used to recrate user sessions), and paradata statistics (which are a group of statistics we derive to learn more about user actions in bulk – the user models do not use these statistics). This image was created by Adam Eck (an IAMAS member) some months ago for a presentation, thanks to him for sharing.

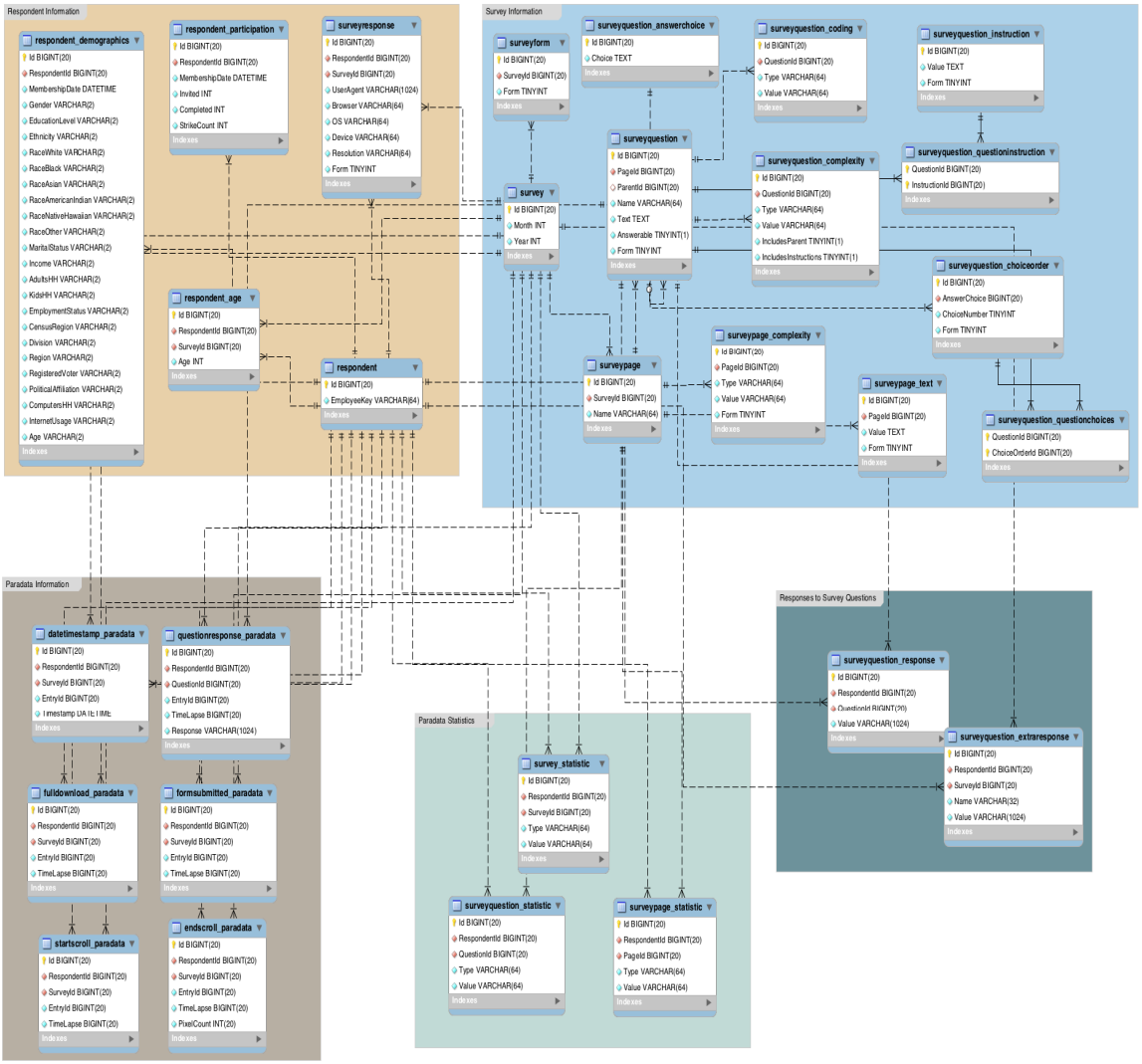

Figure A. Gallup database schema

B. Attributes used in the Gallup panel datasets

Table B below presents the attributes used in the datasets built for the Gallup panel. Six columns appear:

- Attribute: The name and description of the attribute.
- Attribute Type: an indication of whether this attribute is nominal or numeric.
- Subject to Normalization: Indicates whether this attribute is subject to normalization.

- Note that some attributes are clearly behavioral and others are clearly demographics. All behavioral attributes are subject to normalization, demographic attributes are not.

The attributes that are subject to normalization are always normalized in the datasets where the values are normalized to average user behavior or to the current individuals past behavior. However, in the two normalization techniques based solely off current context (by unique page and by page number) some of the behavioral attributes are dichotomized into a total count for the number of actions occurring on the current page, and to the users average amount of those actions per page. This allows us to distinguish between users who are suddenly engaging in a large or small amount of actions on a given page, and those who are displaying a consistent trend in behavior. Each of the behavioral attributes are dichotomized.

Attribute	**Attribute Type**	**Subject To Normalization (i.e. is this a behavioral attribute)**
Number of straight lines	Numeric	Yes
Number of questions answered	Numeric	Yes
Response Time	Numeric	Yes
Scroll Time	Numeric	Yes
Download Time	Numeric	Yes
Number of questions not answered	Numeric	Yes
Page submission time (how long between answering last question and submitting the page)	Numeric	Yes
Number of times user went backwards in the survey	Numeric	Yes
Number of questions viewed by the user	Numeric	Yes
Average time spent per scroll (scrolling through the interface window)	Numeric	Yes

Number of questions skipped (not the same as non-response, a non response means they never answered the question when on the page, all a skip means is that they overstepped a question, though they might have returned an answered it out of order)	Numeric	Yes
Number of scrolls (scrolling through the screen)	Numeric	Yes
Percent of questions answered	Numeric	Yes
Has the user quit early in previous surveys	Nominal	No
Gender	Nominal	No
Education	Nominal	No
Ethnicity	Nominal	No
Race (there is actually one race attribute for each race, but we just present race in this table rather than list them all)	Nominal	No
Marital Status	Nominal	No
Income	Nominal	No
Employment status	Nominal	No
Demographic division (of country)	Nominal	No
Demographic region (of country)	Nominal	No
Registered voter	Nominal	No
Political affiliation	Nominal	No
Computers in house	Numeric	No
Daily internet usage	Numeric	No
Age	Numeric	No
Number of Adults living with respondent	Numeric	No
Number of children living with respondent	Numeric	No
Type of device used in survey (i.e. phone, computer, tablet, etc.)	Nominal	No
Browser used to answer survey	Nominal	No

Operating system used to answer survey	Nominal	No
Did the user commit a rare-action in this survey	Nominal	No

Table B. Attributes in Gallup panel datasets

C. Attributes in ATUS

ATUS attributes are composed of behavioral, demographic, and survey state attributes. The survey state attributes is how we get an idea about user's current context, whereas the behavioral attributes describe user actions, and the demographics attempt to get at intrinsic qualities of users through things like age and employment status. We list the attributes in the Table C, below. The table describes each attribute, indicates if its nominal or numeric, and indicates if it is subject to normalization. Only behavioral attributes are normalized so this indicator also shows what attributes are behavior-based. ATUS is interesting because behavioral attributes can also be state variables – since the state of a survey is a sequence of user behaviors. State variables can also exist independently of behavioral data. Thus we include another column indicating if a variable is a survey state (or context) variable.

Attribute	**Attribute Type**	**Subject To Normalization (i.e. is this a behavioral attribute)**	**Survey state variable**
Education Level of user	Nominal	No	No
Ethnicity	Nominal	No	No
Marital status	Nominal	No	No
Race	Nominal	No	No
Day of week that respondent was interviewed about	Nominal	No	Yes
Indicator of whether the respondent is a	Nominal	No	No

student			
Hours user works per week	Numeric	No	No
User's labor force status	Nominal	No	No
Number of kids user has	Numeric	No	No
Spouse's employment status	Nominal	No	No
Marital status	Nominal	No	No
Time spent alone	Numeric	Yes	Yes
Weekly earnings	Numeric	No	No
Age	Numeric	No	No
Gender	Nominal	No	No
Number of activities performed	Numeric	Yes	Yes
Duration of activities	Numeric	Yes	Yes
28 separate attributes each representing the number of activities performed at given location (ATUS divides where information into 28 categories, rather than list all 28 attributes we note here that we count the number of times a user spent an activity at each location in a separate attribute)	Numeric	Yes	Yes
28 attributes representing the number of activities performed with a given category of people (ATUS divides with whom an activity was performed with information into 28 categories, rather than list all 28 attributes we note here that we count the number of	Numeric	Yes	Yes

times a user spent an activity with each category of people in a separate attribute)			
Number of times activities were edited during interview	Numeric	Yes	No
Number of times activity type (e.g. sleeping vs. eating) was changed	Numeric	Yes	No
Number of activities created during the survey (can be different than number performed if deletes occur or information is changed)	Numeric	Yes	No
Number of activities deleted during survey	Numeric	Yes	No
Number of times the end of the diary was reached (can happen multiple times if mistakes in the diary are made and then corrected)	Numeric	Yes	No
Total number of times hour durations of activities were edited.	Numeric	Yes	No
Total number of times minute durations of activities were edited.	Numeric	Yes	No
Total number of times activity stop times of activities were edited.	Numeric	Yes	No
Total number of activities inserted in between existing activities during a survey.	Numeric	Yes	No
The total number of times the end of the time diary was reached, but then exited as information	Numeric	Yes	No

was changed			
Total number of pop-up windows (dialogue prompts indicating error detection such as a user doing something an unlikely amount of time) appearing	Numeric	Yes	No
Total number of times activity editing changed the state of subsequent activities int eh diary	Numeric	Yes	No
Total time spent editing	Numeric	Yes	No
Total number of edits to activity time information	Numeric	Yes	No
Total number of activity times given as durations (users can say they did an activity x minutes, or until y-clock, a duration means they said they did something x minutes.	Numeric	Yes	No
Total number of activity time information provided by user giving a time the activity stopped.	Numeric	Yes	No
Total number of times the where information was changed for activities	Numeric	Yes	No
Total number of times the with whom information was changed for activities	Numeric	Yes	No
Indicator of if this user had a memory failure during the time diary.	Numeric	Yes	No

Table C. Attributes in the ATUS dataset

D. ATUS session recreation

To recreate user sessions in ATUS log-files (i.e. paradata or audit trail files) are parsed such that survey state changes are found. These state changes are then passed to a survey representation responsible for updating its state appropriately, and accounting for any side effects caused by the state change. It then informs interested observers regarding its state change, among them a separate module responsible for tracking changes to the survey's interface. A tertiary observer watches the state changes of both the survey and interface modules and is used to derive the datasets used in constructing the user models. This task is not accomplished without dealing with several challenges, however.

One challenge is that actions in the log files are communicated such that one line does not convey any real meaning about what the interviewer is doing to change the state of the survey. Instead, multiple lines must be interpreted in their proper context to infer a state change correctly. For example, a single line might communicate a keystroke but does not say where this keystroke is being entered, and thus what it really means. Consequently, we must examine sequences of log entries in order to derive meaning.

Because of this, an application specific parser (i.e. log file and survey specific) had to be developed in order to translate sets of paradata lines into more abstract meanings. This effort was further complicated by the fact that the design of the instrument is such that missing information errors in other parts of the survey affect what is possible to happen via user interaction in later parts of the ATUS survey. This has the unfortunate effect of distorting log-file entry meanings when the survey is in such states. To compensate, we carefully catalogued the states associated with such effects and developed ways to

identify them via the log files. With this in hand, we could then properly interpret the log files in all states of the survey.

Another complication is the appearance of "dialogue lines" in the audit trails, which are indicative that a pop-up box appeared on the user's screen due to the program perceiving some information was missing. The log-file recordings of such "prompts" are often incomplete, conveying neither the user-action which caused the prompt to appear, nor the survey state which necessitated the prompts generation. (For example, one survey state that causes a prompt to appear is if the interviewer records that the user slept for more than 10 hours. If however, the user simply closes the pop-up the log file will not record this fact and will only relate that some prompt appeared). Thus in order to understand all prompts correctly, one must have a complete understanding of the survey state at the point in which the prompt appears. This is important because how the user interacts with a prompt can affect the actual data saved in the survey. Therefore, if we fail to properly track prompts, we will also fail to properly recreate user sessions and thus fail to properly generate data.

Another issue is that the interviewer can insert activities between existing activities at any point. The lines communicating this action, however, are again ambiguous. They fail to distinguish between attempted inserts that failed, and attempted inserts that were actually successfully completed (depending on the current survey state, the program may reject an insert attempt). In fact, successful insertion is dependent on whether all underlying prompts associated with the activity just after the to be inserted activity have been seen and addressed in a manner the program interprets as properly dealing with the issue the prompt alerts the interviewer to. For example, to properly deal with a sleep

prompt, the user must change the time the user slept or confirm that they slept more than ten hours. If the user simply closes the prompt without doing these actions, the prompt will be considered as not dealt with. Then if a user tries to insert an activity just prior to the sleeping activity, the insert will fail. In summary, the only way to know if an insert should be successful is to know the state of the prompts, which are themselves dependent on past inserts. Thus the problem of making sense of these two ambiguous pieces of data is circular.

Due to this difficulty, the developed audit trail parser constructs a combinatorial set of possible inserts and develops a representation of the final survey state based on each possible combination. Some of these combinations fail quickly, as the system doing the parsing and survey state construction finds a contradiction in the survey state – this thus indicates that some insert is being treated as incorrectly failing or succeeding in the current combination of inserts. For other surveys, only one combination is possible (no possible inserts) or only one combination does not fail quickly; and from these we immediately know the correct insert pattern. A third set, yields multiple potential insert patterns following the parsing and state construction. Comparing the final values of the activities captured by the survey state reconstruction, to the final values seen in the public data set further reduces these as we can eliminate insert combinations that yield different final survey states than the state seen in the public data. This reduces the candidates further until most of the surveys have only one viable insert combination pattern remaining. For the final set of surveys that still have the correct insert combination unresolved, a second pass is made through the log files, in which key details of the log-file are examined which potentially indicate whether an insert appeared properly. This

resolves still more ambiguities, until we have only 30 surveys that cannot be resolved in the dataset, likely due to noise elsewhere in the audit trail lines.

The step-by-step process by which surveys are completed gives us a rich source of features that can be pulled out. These include things like the number of times an activity was edited for each piece of information, how long the editing took, and whether activities were deleted or inserted at any point.

It also provides a rich resource to the survey methodology community that they can draw on in manners fitting their ends, as they can ask us to extract specific statistics regarding user interaction of their choosing.

E. Markov Chain Performance When Viewing Entire Surveys

Classification Strategy	Recall	Precision	True Negative Rate	G-mean ($\sqrt{}$ (Recall*TN Rate))
April – Page 4 with Rules	.69	.018	.63	.66
April – Markov Chain	.85	.064	.84	.85
June - Page 10 Rules	.64	.021	.62	.63
June – Markov Chain	.68	.081	.85	.76
September Page 13 Rules	.81	0.0078	0.7529	0.7828
September Markov Chain	.75	.24	.96	.85
November - Page 55	0.69	0.011	0.640	0.663
November Markov Chain	.79	.045	.66	.72
December – Page 9	0.74	0.013	0.685	0.710
December Markov Chain	.67	.034	.64	.65

Table E – Markov Chain Performance on Complete Surveys

Table E shows the performance of the rule-filtering technique on subsequences surveys compared to the performance of the Markov Chain when viewing surveys in their entirety. Rule filtering techniques are labeled with the "Rules" tag, and a page number indicating the discrete point of the survey viewed by the learner. Markov Chain learners are tagged with the survey month they learned from (again, the Markov Chains were trained on entire surveys).

Table E.1 shows the performance of the Markov Chain Technique when learning and classifying whole surveys, versus that of the rule-filtering technique when learning and classifying surveys up to a discrete point. The main difference in the data used by the rule-based technique and the Markov Chain technique is that the following:

- The past technique examined user behavior up to a discrete point in the survey taking sequence.
- The Markov Chain technique examines user behavior across the entire survey.

For example, the "June – Page 10 Rules" classifier examined user behavior up to Survey Page uniquely identified as "Page 10". On the contrary, the Markov Chain technique for the June survey examined the user's behavior over the course of the entire survey.

Precision is poor for both techniques, but Markov Chain Opponent Technique outperforms the subsequence based classifiers in the geometric mean statistic. Breakoff (the rare-action in the Gallup panel) tends to happen early in surveys, and thus one explanation for the boost in performance achieved by the Markov Chain viewing the whole survey is that it could begin to distinguish this length difference due to the

consequent changes in transition probabilities that accrued due to differences in the amount and nature of the questions answered.