Technical Report

Department of Computer Science
and Engineering
University of Minnesota
4-192 Keller Hall
200 Union Street SE
Minneapolis, MN 55455-0159 USA

TR 14-015
Formerly TR 13-033

Revised: A Hazard Based Approach to User Return Time Prediction

Komal Kapoor, Mingxuan Sun, Jaideep Srivastava, and Tao Ye

July 23, 2014

A Hazard Based Approach to User Return Time Prediction

Komal Kapoor
Department of Computer Science
University of Minnesota
Minneapolis, MN 55455
kapoo031@umn.edu

Mingxuan Sun
Pandora Media Inc.
2101 Webster Street
Oakland, CA 94612
msun@pandora.com

Jaideep Srivastava
Department of Computer Science
University of Minnesota
Minneapolis, MN 55455
srivasta@cs.umn.edu

Tao Ye
Pandora Media Inc.
2101 Webster Street
Oakland, CA 94612
tye@pandora.com

ABSTRACT

In the competitive environment of the internet, retaining and growing one's user base is of major concern to most web services. Furthermore, the economic model of many web services is allowing free access to most content, and generating revenue through advertising. This unique model requires securing user time on a site rather than the purchase of good which makes it crucially important to create new kinds of metrics and solutions for growth and retention efforts for web services. In this work, we address this problem by proposing a new retention metric for web services by concentrating on the rate of user return. We further apply predictive analysis to the proposed retention metric on a service, as a means for characterizing lost customers. Finally, we set up a simple yet effective framework to evaluate a multitude of factors that contribute to user return. Specifically, we define the problem of return time prediction for free web services. Our solution is based on the Cox's proportional hazard model from survival analysis. The hazard based approach offers several benefits including the ability to work with censored data, to model the dynamics in user return rates, and to easily incorporate different types of covariates in the model. We compare the performance of our hazard based model in predicting the user return time and in categorizing users into buckets based on their predicted return time, against several baseline regression and classification methods and find the hazard based approach to be superior.

Categories and Subject Descriptors

J.4 [**Computer Applications**]: Social and Behavioral Sciences; H.3.5 [**Online Information Services**]: Web-based services

General Terms

Human Factors, Measurement, Experimentation

KDD'14, August 24–27, 2014, New York, NY, USA.

http://dx.doi.org/10.1145/2623330.2623348 .

Keywords

online user behavior, customer relationship management, growth and retention, hazard based methods

1. INTRODUCTION

User attention is perceived as the most important resource in the internet era [12]. The web is described [23] as a *'virtual theme park where most rides are free such that revenue is generated through "selling eyeballs" to advertisers'*. The ad-supported economy of the web has the web-services vying for users' time rather than their money. Having a large loyal and dedicated user base has several indirect benefits as well. Many services grow with their users, improving themselves based on their feedback and through the power of big data analytics on their activities logs. A common example is the Google search engine, which has perfected its query auto-correct feature primarily using user click-through data, as well as improved its search performance regularly using user search histories. Furthermore, an active community can be tapped to create new content that benefits the other users of the service and the service as a whole as seen for popular social networks such as Facebook and Twitter.

There is tremendous competition among the rapidly increasing number of web services for the finite and limited resource corresponding to user attention. Although, attracting new customers is crucial for any business, it is generally much easier and cheaper to retain existing customers [4, 29]. This directly results in a great deal of emphasis being placed on engaging one's current user base. Customer retention efforts have been heavily researched in sectors such as telecommunication [29], financial services [22], internet services [13] and other utilities etc. which follow the subscription based model. The methods in these domains have focused on identifying potential churners in the user population, where churners are defined as those current subscribers who are not likely to renew their subscription in the coming months. Once detected, the churner population is targeted with retention strategies like offers, customer solutions and recommendations to win them back.

However, such methods cannot be directly applied to solving the user retention problem for web services due to the following reasons:

1. **Difficult to define churn for a non-contractual setting.** A non-contractual business such as a free web service, does not receive a definitive indicator of termination from the user. To counteract this problem, some alternative definitions of churn have been proposed. Churn is defined as a significant drop in a user activity levels [16]. Another work addresses this problem by first providing a definition for a loyal user of a service and then defining churners as those users who were loyal to the service but are no longer so [21]. However, such methods remain sensitive to changes in their proposed definition of churn.

2. **Highly dynamic user visitation behavior.** Web services offer none or negligible switching costs to users. With no financial commitments towards a service, users switch quite frequently between different services. The highly dynamic nature of user visitation behavior makes it difficult to define typical activity volumes for a user and to segregate users as active and inactive with respect to the service.

To adapt to the unique incentive structures and dynamic user base, in this work we propose a novel retention metric which tracks the user return rate for addressing growth and retention in web services. The user return rate is defined as the fraction of the existing users returning to the service on a particular day. It is beneficial for a web service to improve its user return rate in order to increase its revenue. Predictive analysis of user return times can direct such improvements efforts. Return time prediction allows a service to identify indicators of earlier (longer) return times for their users. Identifying such indicators and quantifying their impact on user return times offers a service insights into its practices. It also enables a service to employ corrective measures and improve the experience to its users. Secondly, a service can identify sections of its user base that are not likely to return soon. Studies have shown that the longer the users stay away from a service, the less likely are they to return in the future [24]. Early identification of users who are not likely to return soon to the service allows the deployment of suitable marketing strategies to encourage those users to engage with the service again.

We propose a hazard model [8] from survival analysis to predict user return times. The hazard based models are preferred over the standard regression based methods for this problem due to their ability to model particular aspects of duration data such as censoring. More importantly, the Cox's proportional hazard regression model is used as it can incorporate the effects of covariates [1]. We apply the model for return time prediction on real-world datasets from two popular online music services.

We now summarize the key contributions made by us in this paper:

(a) We formally define an approach for targeting retention solutions in free web services via user return time prediction.

(b) We propose the Cox's proportional hazard to model dynamic return events and incorporate the effects of covariates for return time prediction. We develop useful return time predictors and conclude correlations between user usage patterns and their return times.

(c) The Cox's proportional hazard model outperforms state-of-the-art baselines in both return time prediction and user classification based on predicted return time.

The rest of the paper is organized as follows. In **Section 2** we provide a brief overview of the related research in the area of churn prediction and the use of hazard based methods. We then formally define our problem and lay out our contributions in **Section 3**. In **Section 4** we describe our hazard based predictive model and provide details of the covariates used and the model estimation procedure. In **Section 5** and **Section 6** we discuss the experimentation setup and the results. We summarize the conclusions from our experimental analysis and provide future directions for our work in **Section 7**.

2. RELATED WORK

In this work, we propose a new approach for directing growth and retention solutions for web services through return time prediction. Traditionally studies have focused on the problem of churn prediction defined as a binary classification problem where users are categorized based on several behavioral and demographic features into two categories: future churners or non-churners. The popular data mining techniques used for building classifiers for churn prediction include decision trees such as CART and C4.5 etc. [29], logistic regression [4], support vector machines [7] and neural networks [28], though random forests [7, 30] are found to be better in performance. Ensemble methods have been used to combine multiple classifiers to construct powerful meta-classifiers and to handle the class imbalance problem typical to churn prediction [5]. Alternatively, approaches from survival analysis have been used to model the dynamics in the churn event rate with user tenure [14]. The churn event rate for users is found to decline with tenure such that new users are more likely to churn than tenured users.

A major hurdle to applying these methods to free-to-use services discussed in this paper is to provide an appropriate definition of churn. Studies on user lifetime modeling and retention for online environments have used different criteria for defining the loss of a customer, such as the period of inactivity [31], decrease in activity [16] or indirectly, via a definition of loyal users [21], discussed earlier. Yang et al. [31] have further studied how user participation patterns affect the length of their lifetimes on online knowledge sharing communities. However, most of these methods focus on the length of user participation rather than the volume of their activities. Instead, online businesses are increasingly paying attention to their returning users rather than the count of their registered users. Further, the research community has started concentrating on analyzing and modeling users activities on different types of websites [18, 33]. A major focus of these methods have been to understand how websites memberships, specifically measured in the number of active users, evolves with time and correlate such factors to the success or failure of the website [25]. Also, many studies on the measurement and improvement of intra-site [2, 17, 10] and inter-site engagement have emerged [32]. Many of these studies identify return time as a robust metric for user engagement. All these factors suggest that continuous tracking and improvement of user engagement, measured in

[1]We use the term covariates to describe features or predictors in our model. The choice of terminology is based on that used in the survival analysis literature from where we adopt our model.

terms of their return time, is crucial for the performance goals of web services. Hence, in this work we directly focus on the return time metric for organizing retention efforts for web services. We use a Cox's proportional hazard regression model [8] from survival analysis for this problem as the model can quantify the impact of covariates on the target event rate. This unique property results in the Cox's Model being a popular choice for several online user behavior studies [10, 31]. Additionally, we also define different types of return time predictors suitable for this problem.

Several types of covariates have been used for churn prediction. RFM models [11] propose the use of three variables, Recency, Frequency and Monetary value of their previous interaction for identifying potential churners. Other covariates based on demographics, contractual details, service logs, use patterns, complaints and customer service responses [3, 29] have been found useful. We use some of these covariates in our model. In addition, we also incorporate user behavior related covariates in order to understand how user interactions while engaging with the service affect the rate of their return in the future. A special feature of our model is that it can handle the recency variable implicitly by computing the expected future time of return for the users given their length of absence from the service.

3. RETURN TIME PREDICTION FOR WEB SERVICES

A user's visitation behavior on a free web service tends to be quite flexible and arbitrary post registration partially due to the lack of financial investments and constraints. Instead, the length of the tenure of users of web service displays a power law distribution with most of the users never returning back to the service [9]. In this work, we adopt a unique methodology for analyzing the dynamic user visitation data by directly modeling the user return time.

3.1 Problem Statement

We define users as belonging to either of the two activity states - the *in* and the *out* states. When users visit the service, they are said to be in the *in* state; while, when they do not visit the service, they are said to be in the *out* state.

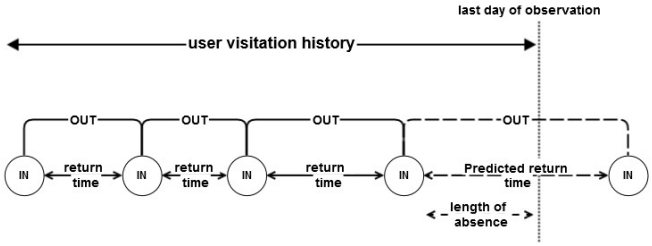


Figure 1: State Space Diagram

We focus on the problem of predicting the return time of the users which is the time the user spends in the *out* state. The return time for a user can potentially extend to infinity (for users who never return back to the service). Therefore, a threshold, t_d is defined on the return time and we predict the return time for the users up to time t_d. The return time prediction problem may be formally defined as follows:

Definition 1 Given that the last time the user was in the *in* state was at time t_0, the return time prediction problem is to predict the quantity $min(t_r, t_d)$, also called the truncated return time (T_{rd}), where t_r is the total time the user spends in the *out* state and ranges from 0 to ∞, t_d is a finite threshold on the return time and either of the following holds:

(a) the user is expected to return to the *in* state at time $t_0 + t_r$, if $T_{rd} = t_r$, or

(b) the user is expected to stay in the *out* state for at least t_d units of time, if $T_{rd} = t_d$

Figure 1 provides a diagrammatic representation of the user return time prediction problem.

3.2 Time Dependence in User Return Time

The time duration between events has been studied extensively in queuing theory, for example to study the waiting time between customer arrival and customer service events. Such events are commonly modeled to generate from a Poisson process such that the waiting times follow the exponential distribution. An attractive property of the exponential distribution is the memoryless property which entails that the future rate of occurrence of the event is independent of the elapsed time. For a random variable T denoting the time of occurrence of the event, the following equation is said to hold if the memoryless property is satisfied:

$$P(T > t + s|T > s) = P(T > t) \quad (1)$$

However, several phenomena are seen to defy the simple memoryless property in interesting ways. For example, the rate of adoption of new products is found to increase with the elapsed time [26]. Alternatively, the rate of events like responses to surveys, promotions [15] etc. is seen to decline with the elapsed time. The decline in future event rate with the elapsed time, has been referred to as *'inertia'*. We suspect similar type of inertia in user return behavior. For duration data showing time dependence, it becomes meaningful to compute the expected future time of the event given the elapsed time, $E(T|T > s)$. We, now define the problem of predicting the expected future time of return of the users given their length of absence (LOA) from the service.

Definition 2 Given that the last time the user was in the *in* state was at time t_0, and he has already been in the *out* state for time t_s, the future return time prediction problem is to predict the quantity $min(t_{fr}, (t_d - t_s))$, also called the truncated future return time T_{frd}, where t_{fr} is the additional time the user spends in the *out* state and ranges from 0 to ∞, t_d is a finite threshold on the return time and either of the following holds:

(a) the user is expected to return to the *in* state at time $t_0 + t_s + t_{fr}$, if $T_{frd} = t_{fr}$, or

(b) the user is expected to stay in the *out* state for atleast $t_d - t_s$ more units of time, if $T_{frd} = t_d - t_s$

4. METHOD

We consider a time window over which user return time observations are collected. Each return time observation can be associated with a set of covariates influencing its magnitude. Hence, the data can be represented as a set of tuples:

$< X, T >$ where, T is the return time observation and X is the vector of covariates associated with that observation. Since a user can return to the service multiple times during the considered time window, we can have multiple tuples corresponding to a single user.

There are two aspects of the collected data that need special attention.

1. Censoring: Duration data which is collected over a fixed time period tends to have incomplete observations corresponding to events which were yet to happen at the end of the study period. Such observations are said to be censored and this particular type of censoring is called right censoring. In order to capture censored observations as well, a special variable *status* is added to the representation of duration times. The *status* variable is set to 0 when the time variable represents the actual observation of return time whereas it is set to 1 when the time variable represents a censored observation. In the latter case the time duration represents the time gap between the user's last visit and the end of the study period. Ignoring censored observations biases one's analysis towards earlier returns. A major advantage of the hazard based methods is that they can handle censored observations quite well.

2. Recurrent observations: The collected data may contain more than one return time events, also called recurrent events, per user during the study period. The active users have many more return time observations than inactive users. Retaining these observations can bias our analysis towards the active users which is detrimental to our objective of targeting losing customers. However, we lose information if we throw away these observations. Instead, we use a simple weighting scheme for handling recurrent events. We weight each observation corresponding to a user with the inverse of the number of observations made for that user. Hence, each user has a unit weight in the data but we incorporate all observations made for him/her. Later in the paper, we discuss the sensitivity of our results to this weighting scheme. Some care needs to be taken while testing models when working with recurrent data and we discuss that in our *Experiments* section.

4.1 Hazard Based Prediction Model

Survival analysis is a branch of statistics which deals with the time of occurrence of events, also called duration modeling. It offers a rich set of methods which allow us to easily address questions like what is the probability that an event is going to happen after t units of time or what is the future rate of occurrence of the event given it has not happened in t units of time. In this work we deal with discrete measures of time. Two functions are useful for analyzing duration information:

The survival function at time t is defined as:

$$S(t) \;=\; P(T > t) \tag{2}$$

where T is a random variable denoting the time of occurrence of the event. The hazard function measures the instantaneous rate of occurrence of the event at time t, conditioned on the elapsed time t.

$$\lambda(t) \;=\; P(T \;=\; t \mid T \geq t) = -S'(t)/S(t-1) \tag{3}$$

The Cox's proportional hazard model is commonly used to incorporate the effect of covariates on the hazard rate. The model is based on the simple assumption that the covariates affect the magnitude of individual hazard rates but not the shape of the hazard function. Expressed mathematically,

$$\lambda(t) = \lambda_0(t) * \exp(\beta_1 * X_1(t) + \beta_2 * X_2(t) + ...) \tag{4}$$

where, λ_0 is the baseline hazard function, $X_1(t)$, $X_2(t)$, etc. are the covariates which may be static or may vary with time and β_1, β_2 etc. are the regression coefficients. The ability of the Cox's model to handle time-varying covariates is an important feature of the model.

One can obtain the survival function from the hazard function using the following equations:

$$\Lambda(t) = \sum_{0}^{t} \lambda(u)\, du, \tag{5}$$

$$S(t) = \exp(-\Lambda(t)), \tag{6}$$

where Λ is defined as the cumulative hazard function. The expected time of return can then be computed using the equation below:

$$E(T) = \sum_{0}^{\infty} S(t). \tag{7}$$

Furthermore, the expected future time of return given the time not returned for (t_s) can be computed as follows:

$$E(T|T > t_s) = \frac{1}{t_s} \sum_{t_s}^{\infty} S(t). \tag{8}$$

The survival function is truncated beyond a point of time or when the probability of survival drops below a threshold in order to prevent the return time estimate from diverging. For our prediction problem, we impose t_d as an upper bound on the return time estimate. Hence, the expected return time and the expected future return time computations can be re-defined as:

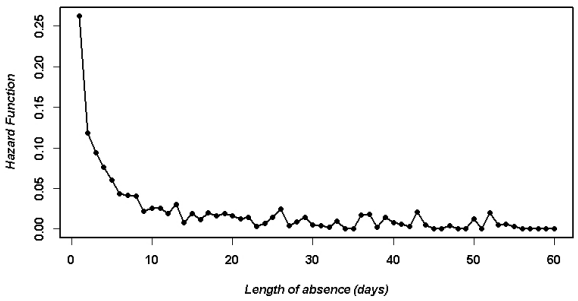
$$E(T) = \sum_{0}^{t_d} S(t), \tag{9}$$

$$E(T|T > t_s) = \frac{1}{t_s} \sum_{t_s}^{t_d} S(t). \tag{10}$$

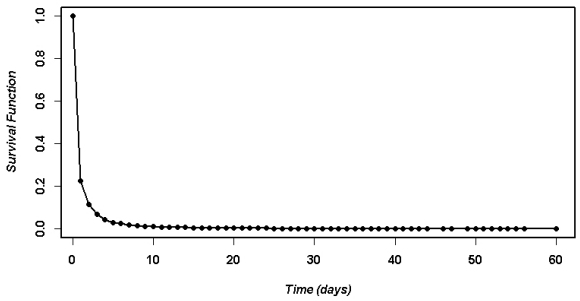
4.2 Model Estimation

The Cox's proportional hazard model is a semi-parametric model as it does not assume a mathematical form for the baseline hazard function. Instead, the model can be broken down into two factors. The first factor represents the effect of the covariates on the hazard rate. The effect parameters (regression coefficients) are learnt by maximizing the partial likelihood which is independent of the baseline hazard function. Once the regression coefficients have been learnt, the non-parametric form of the baseline hazard function is estimated using the Breslow's method. Cox's seminal paper [8] is a good reference for the details of the estimation procedure.

We use the standard survival package in R for estimating the Cox's model. The survival package can handle weighted data instances. We use days as the unit of time for our analysis as most of the users in our datasets are found to return



(a) Baseline Hazard Function



(b) Survival Function

Figure 2: The baseline hazard function and the survival function computed on the Last.fm training dataset.

within the first week. A user is considered to have visited the service on a day if he performed at least one activity on the service on that day. One may define more stringent criteria on user activity for this purpose, such as minimum time spent, number of interactions etc. The threshold (t_d) for the return time prediction problem is set to 60 days, which is a reasonably large value and beyond which users are already the focus of retention efforts. Return time observations larger than 60 days are hence assumed to be censored. In our experiments, we also study the performance of the model for different choices of the threshold.

5. EXPERIMENTAL SETUP

We now evaluate the performance of the Cox's proportional hazard model for solving our proposed return time prediction problem. We consider both the performance of the model at predicting the return time of the user and at classifying users based on their expected return times. Such a categorization procedure is the logical next step for a service looking to target marketing strategies to users not likely to return soon. For both the problems, we also evaluate how well the Cox's model can incorporate the LOA information by re-estimating the expected future return time given the LOA.

5.1 Data Collection

For our experiments we use a small public and a larger proprietary dataset. We briefly describe these two datasets:

- The Last.fm dataset. Last.fm, is an online music website catering to millions of active users. Recently, the service made available the complete music listening histories of around a 1000 of its users as recorded until May, 2009 [1, 6]. For every song the user listened to, the dataset includes the song title, the artist name and the timestamp at which the song was heard. We use two separate time windows for creating the training and the testing datasets. All user visits observed during Oct, 2008 - Dec, 2008 were used to test the model through cross-validation. We also tested our model on future user visits observed from Jan, 2009 - Mar, 2009.

- Large-scale dataset. Our proposed approach was applied as a part of the growth and retention efforts for a large ad-supported music service. A dataset of around 73,465 users, collected over 3 months from May, 2012 - July, 2012, was used for training and testing our model via cross-validation.

5.2 Covariates

We constructed the following covariates for the return time prediction problem.

- **Covariates related to the typical visitation patterns of a user**. Such covariates seek to predict the future return behavior of the users based on how their visitation behavior has been historically. For example, users who have been highly frequent in the past (loyal to the service) are likely to remain frequent in the future and similarly users who have been infrequent in the past (casual visitors) are likely to visit infrequently in the future.
  - Active Weeks: This covariate is defined as the ratio of the number of weeks since registration when the user visited the service at least once to the total number of weeks elapsed since registration.
  - Density of Visitation: This covariate captures the volume of user activity on the service for the weeks the user is active on the service. It is defined as the average number of days the user visited the service during the weeks the user visited the service at least once.
  - Visit Number: This covariate is used to measure how tenured the user is with the service.
  - Previous Gap: This covariate represents the most recent return time observation (which is the gap between the user's last and prior to the last visit) for the user. For first time users this covariate is set to -1.
  - Time weighted average return time (TWRT): This covariate measures the average return time for a user. The return times are further weighted by the inverse of the length of time elapsed since they were observed under the premise that the more recent return times are more informative about the user's current visitation behavior.
- **Covariates related to user satisfaction and engagement with the service**. Satisfaction and engagement related covariates are more difficult to construct as they attempt to capture latent user emotions

about the service. Such can be extracted from any explicit (likes, dislikes, complaints etc.) or implicit (time spend, unique activities etc.) feedback indicators using user past interactions. In this work, we constructed these covariates based on user activities recorded on the last visit to the service (last *In* state)

  - Duration: This covariate captures the time spend at the service measured by the number of songs heard by the user.
  - % Distinct Songs: This covariate measures the fraction of the number of distinct songs listened by the users over the total number of songs listened by them.
  - % Distinct Artists: This covariate measures the fraction of the number of distinct artists listened by the users over the total number of songs listened by them.
  - % Skips: This covariate measures the fraction of the number of songs skipped by the users of the total number of songs listened by them. The skip information is not directly available for the Last.fm dataset. Instead, we indirectly identified skips by comparing the gap between two consecutive songs (s_1 and s_2) in the data with the length of the song s_1. If the time gap was found to be less than the length of song s_1 by more than 30 seconds, then song s_1 was identified to have been skipped. The API, track.getInfo made freely available by Last.fm was used to retrieve the duration for the songs in the dataset.
  - Explicit feedback indicators: These covariates include information obtained directly from the users such as ratings, comments, complaints etc. Explicit feedback measures tend to be highly accurate and are an important source of information about user's satisfaction with the service. However, they are hard to acquire as providing explicit feedback requires user effort. We did not have any explicit feedback indicators for the Last.fm dataset. We had such ratings for our proprietary dataset which were included in the model.

- **Covariates used for abstracting the effects of external factors**. External factors include public holidays and weekends, marketing campaigns and promotions or personal factors which impact the rate of user return. The ability to model external factors is very useful as by modeling these covariates, we can both quantify the impact of these factors and control for these effects to improve our analysis on the other covariates. For simplicity, we have not considered any external covariates in our experiments. However, in our *Conclusion* section, we discuss how the Cox's model can be used to model the day of the month covariate which allows us to incorporate weekly effects and holiday effects in our predictions.

5.3 Evaluation Metrics and Baselines

Different baselines are used for evaluating the performance of the Cox's model at the regression and the classification tasks. All baselines are implemented using the same covariates as used in the Cox's model. For the regression problem we compared the Cox's model against simple average (trivial baseline), linear regression, decision tree regression (RepTree), Support Vector Machine (with linear kernel) and neural networks (multilayer perceptron). Support Vector Machine Regression took too long to run (more than a day) on our large scale dataset and was omitted in those results. The performance of the models were evaluated using Weighted Root Mean Square Error(WRMSE). The WRMSE is computed by weighting the error between the true return time and predicted return time with the weight of the test instance as follows:

$$WRMSE = \sqrt{\frac{\sum_{i=0}^{N} w(i) * (T^p_{rd}(i) - T_{rd}(i))^2}{\sum_{i=0}^{N} w(i)}} \tag{11}$$

where, N is the number of test instances, $T^p_{rd}(i)$ denotes the truncated return time predicted for the i-th observation and $T_{rd}(i)$ denotes the true truncated return time the i-th observation. We can replace $T^p_{rd}(i)$ with $T^p_{frd}(i)$ and $T_{rd}(i)$ with $T_{frd}(i)$ for computing the WRMSE for the expected future return time predictions.

Our classification baselines included logistic regression, random forest, support vector machine (with a linear kernel) and neural networks (multilayer perceptron). We used weighted F-measure for the minority class for measuring performance at the classification task. The weighted f-measure is defined as the harmonic mean of the weighted precision and weighted recall scores which are defined as follows. The set A denotes the instances actually belonging to the minority class and set P denotes the instances which were predicted to belong to the minority class.

$$\text{Weighted Precision} = \frac{\text{sum of weights of instances in } A \cap P}{\text{sum of weights of instances in } P} \tag{12}$$

$$\text{Weighted Recall} = \frac{\text{sum of weights of instances in } A \cap P}{\text{sum of weights of instances in } A} \tag{13}$$

The experiments for the baselines were conducted using Weka, the open source data mining software available under the GNU General Public License. The baselines were suitably tuned using a hold out set. Also, Weka provides support for handling weighted data instances allowing us to easily incorporate the weight vector while training the models. Since a direct application of cross-validation would not maintain temporal ordering between observations of the same user, for our evaluation we took special care to ensure that all recurrent data corresponding to a user belonged to the same fold. This was done by first randomly dividing users into different folds and then placing all observation associated with the user in that fold. As a result, the training and testing folds had observations from different users.

6. RESULTS

In this section we analyze the results of the experimental evaluation of the Cox's model.

6.1 Model Parameters

We only discuss the parameters of model trained on the Last.fm dataset.

The importance of the covariates for the prediction problem can be assessed using different importance indicators (Table 1). The regression coefficients and the significance

score for the covariates can be obtained directly from the output of the R function for fitting the Cox's model. The regression coefficient tells us how much a unit change in the value of the covariate impacts the user's rate of return. For example, with every song the users listened during their last visit, their hazard rate was found to multiply by $\exp(1.315e-03) = 1.0013$, decreasing their return times estimates. The value of the coefficient was statistically significant at a significance level of 0.01. We found most of the covariates associated with the typical patterns of visitation (Active Weeks, Density, Previous gap) to be highly significant for predicting the return time variable. Also, some of the engagement/satisfaction related covariates, namely duration and % artists had significant effects on the hazard rate. We also computed the mean of the product of the covariate and its coefficient (MEAN$(X * \beta)$) measured for all instances in the training set. This provided an average score for how much the covariate impacted the magnitude of the baseline hazard function. The density covariate impacted the rate of return the most on an average for our dataset.

Covariates	**Coefficient**	**Significance**	**MEAN**$(X*\beta)$
Active Weeks	9.313e-02	2.140e-02*	4.370e-01
Density	2.366e-01	1.050e-13***	1.244e00
Visit Number	4.941e-05	7.318e-01	2.336e-02
Previous Gap	-5.175e-03	1.470e-03**	-1.222e-02
TWRT	-1.484e-02	2.817e-01	-2.492e-02
Duration	1.315e-03	2.538e-02 *	6.171e-02
% **Distinct Songs**	6.849e-02	7.653e-01	6.040e-02
% **Distinct Artists**	-2.251e-01	8.553e-02 .	-1.064e-01
% **Skips**	3.740e-01	2.322e-01	4.873e-02

Table 1: Covariate Importance Indicators for the Last.fm Dataset. Signif. codes: 0 '*' 0.001 '**' 0.01 '*' 0.05 '.'**

Figure 2 displays the baseline hazard function and the survival function computed for the training dataset from Last.fm. The baseline hazard function has a sharply declining shape typical of processes exhibiting inertia. Hence, the longer users stay away from the service the lesser likely they are of returning within sometime in the future. As a result, it is all the more important for a web service to ensure that its user are motivated to return to the service soon. The survival function has a value of 0.0009 at 60 days. This suggests that 0.09% of users for this dataset did not return within 60 days.

6.2 Return Time Prediction

Table 2 and Table 3 display the weighted root mean square error scores obtained using the hazard based approach and the standard regression based approaches for the large-scale proprietary and the Last.fm datasets, respectively. We find that the hazard based approach is superior in predictive performance than the other baselines and the improvements are highly significant (p-value$< 10^{-10}$, using two-tailed paired t-test). The hazard based approach also fares well in terms of run time. On a Intel(R) Xeon(R) CPU X5650 @ 2.67GHz 24GHz, the hazard based approach takes $\sim$ 8 minutes as compared to neural networks which take $\sim$ 16 minutes to finish one run of cross-validation. Decision tree regression ($\sim$ 4 minutes), linear regression ($\sim$ 26 seconds) and average ($\sim$ 20 seconds) are faster however, the lower run times come at the cost of performance.

As discussed earlier, the hazard based approach allows us to compute the expected future return time for a user given their length of absence (LOA) by incorporating the dynamics in the hazard function. We evaluate the performance of the hazard-based approach in updating its prediction given the LOA values. Since the standard regression approaches do not provide similar functionality, we re-learn those models by incorporating the LOA values as a separate feature. The values for this feature is generated by replicating each return time observation T, T times for all values of LOA ranging from $(0) - (T-1)$. The future return time is appropriately reassigned to range from $(T) - (1)$. Doing so can significantly increase the size of the dataset. The data instances are re-weighted to ensure that each user still holds a unit weight in the test and the training sets. Due to space limitations we only show the comparisons between two of our baselines: decision tree regression (best performing baseline) and linear regression (because of its ease of use), with the hazard based approach for the large-scale proprietary dataset. We find that the hazard based approach is superior than both these models in estimating the expected future return time (Fig.3).

	Training Data (10-fold Cross Validation)
Average	19.41
Linear Regression	18.54
Decision Tree Regression	18.14
Neural Networks	18.26
Hazard Based Approach	16.58

Table 2: WRMSE for User Return Time Prediction using the Proprietary Dataset.

	Training Data (10-fold Cross Validation)	**Test Data**
Average	10.55	10.40
Linear Regression	9.61	9.37
Decision Tree Regression	9.45	9.15
Support Vector Machine	10.76	10.33
Neural Networks	9.58	9.36
Hazard Based Approach	8.76	8.45

Table 3: WRMSE for User Return Time Prediction using the Last.fm Dataset.

6.3 Classification into User Buckets

The users are classified into different categories based on their predicted return times. For the Last.fm dataset we bucketed users based on their predicted return times being larger or within 7 days, while for the larger proprietary dataset we classified them based on their predicted return times being larger or within 30 days. The shorter time period was used for the Last.fm dataset due to scarcity of users in the test set that returned after 7 days. Table 4 and Table 5 provide the performance scores for the hazard based approach and the other baselines for classifying instances into the minority class for the proprietary and the Last.fm datasets. The proprietary dataset had 15.4% samples and the last.fm dataset had 12.2% samples belonging

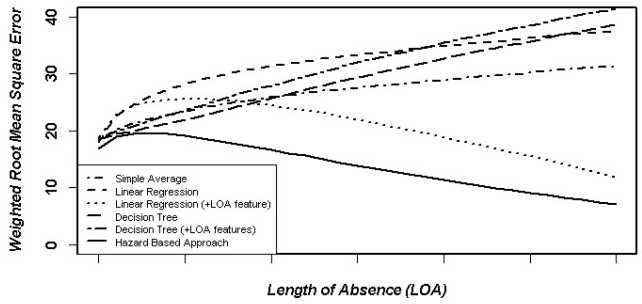


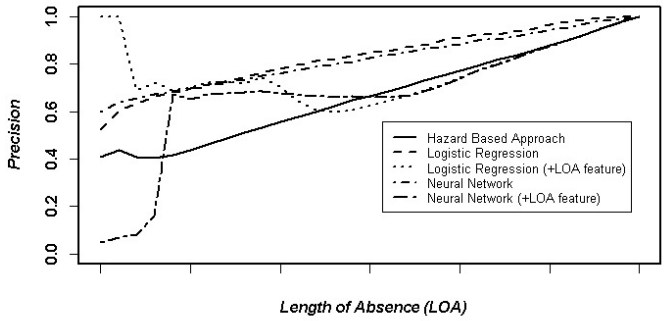
Figure 3: WRMSE for different values of LOA for the Proprietary Dataset. The units on the X-axis have been omitted.

	Training Data (10-fold Cross Validation)		
	Precision	Recall	F-Measure
Random Forest	0.47	0.10	0.18
Logistic Regression	0.52	0.08	0.15
Support Vector Machine	0	0	0
Neural Networks	0.48	0.17	0.25
Hazard Based Approach	0.41	0.23	0.29

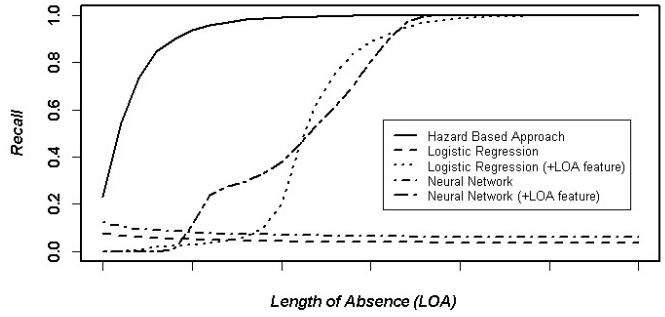
Table 4: Weighted precision, recall and f-measure scores for the minority class (expected return time > 30) for the Large-scale Proprietary Dataset.

to the minority class. A naive classifier would have a precision of 0.154 and 0.122, respectively for these datasets. All the models perform better than a naive classifier. Although, the hazard based model is not learnt as a classification model, it still performs superior to the state-of-the-art baselines for our proprietary dataset (p-value$< 10^{-8}$, using two-tailed paired t-test) and is comparable in performance to the best performing baselines for our Last.fm dataset. The hazard based approach has the highest recall of all the models which seems to be at the cost of precision. However, for a rare class problem like ours, recall at identifying most of the at-risk users is far more important to a business and the overheads from the lower precision are low. In terms of run time, on a Intel(R) Xeon(R) CPU X5650 @ 2.67GHz 24GHz, the hazard based approach takes ~ 8 minutes to finish one run of cross-validation, which is lower as compared to the other baselines: neural network classifier (~ 15 minutes), logistic regression (~ 11 minutes) and support vector machine (~ 24 minutes). Random forest has the lowest run time of all the models (~ 6 minutes).

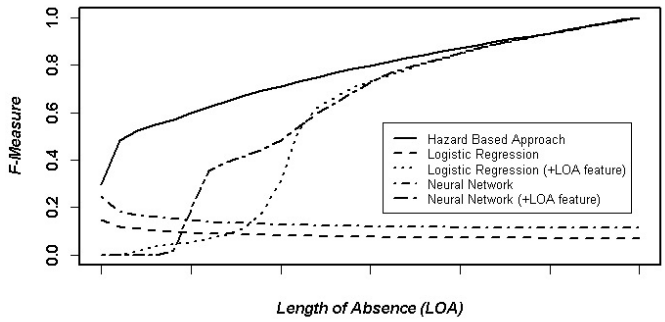
We also evaluate the performance of the hazard based approach in classifying users into buckets given the LOA values. Again, the classification baselines do not offer similar capabilities for updating their prediction scores given LOA values. Hence, we incorporate LOA values as an additional feature for classification and replicate instances to populate the values for the feature as done for the standard regression methods earlier. We provide comparison results against the best performing baseline classification approaches - logistic regression and neural networks. We find that the hazard-based approach can incorporate the LOA information and update its prediction much effectively as compared to both logistic regression and neural networks (Fig. 4).



(a)



(b)



(c)

Figure 4: Figures (a), (b) and (c) are the plots of the weighted precision, recall and f-measure scores respectively, for different values of LOA for the Large-Scale Proprietary Dataset. The units on the X-axis have been omitted.

6.4 Sensitivity to the Threshold

The threshold (t_d) was set to 60 days in our experiments, which was a reasonably large value and beyond which users are already the focus of retention efforts. For completeness, we also evaluate our model for some smaller values of the threshold. Table 6 lists the performance of the models at predicting the return time for threshold values of 15, 30 and 45 days. We find that the Cox's model still performs

	Training Data (10-fold Cross Validation)			Test Data		
	Precision	Recall	F-Measure	Precision	Recall	F-Measure
Random Forest	0.64	0.24	0.35	0.72	0.29	0.41
Logistic Regression	0.68	0.44	0.53	0.66	0.40	0.50
Support Vector Machine	0.61	0.11	0.18	0.82	0.15	0.25
Neural Networks	0.77	0.39	0.52	0.71	0.36	0.48
Hazard Based Approach	0.39	0.79	0.52	0.37	0.81	0.51

Table 5: Weighted precision, recall and f-measure scores for the minority class (expected return time > 7) for the Last.fm Dataset.

better than the other baselines at the prediction task in these experiments (p-value$< 10^{-8}$, using two-tailed paired t-test).

	Training Data (10-fold Cross Validation)		
	$t_d = 15$	$t_d = 30$	$t_d = 45$
Average	6.45	11.77	16.07
Linear Regression	6.11	11.16	15.29
Decision Tree Regression	5.14	10.11	14.61
Neural Networks	5.29	10.36	15.28
Hazard Based Approach	5.04	9.54	13.41

Table 6: WRMSE for User Return Time Prediction with different values of t_d using the Proprietary Dataset.

6.5 Alternative Approaches for Handling Recurrent Observations

We use a re-weighting scheme for handling recurrent observations which allows us to retain all data instances for a user in the dataset without biasing the models towards active users. However, we now evaluate the sensitivity of our results to our weighting schemes by considering alternative approaches for handling recurrent observations. Four such approaches are defined: unweighted, using only the first observation per user, using only the last observation per user and considering only users active on a particular date chosen randomly. The last three approaches eliminate recurrent observations by data selection. We use Root Mean Square Error (RMSE) for evaluation. Due to space constraints we only report RMSE results on our proprietary dataset.

	Training Data (10-fold Cross Validation)			
	Un-weighted	First Event	Last Event	Single day
Average	7.62	17.35	26.17	7.44
Linear Regression	7.33	16.80	24.96	7.08
Decision Tree Regression	7.37	16.52	24.56	6.99
Neural Networks	7.31	17.42	24.52	7.01
Hazard Based Approach	7.31	15.955	17.76	6.87

Table 7: RMSE for User Return Time Prediction with alternative schemes for handling recurrent observations using the Proprietary Dataset.

We find that the Cox's model outperforms the other baselines when we use only the first or the last observation per user for training and testing the models (p-value$< 10^{-10}$, using two-tailed paired t-test). All the models have comparable performance when we use the un-weighted scheme or work with user observations recorded on a particular day. Both these scheme also record the lowest errors compared to the other schemes for all the models. We suspect this to happen because both these schemes are dominated by the active users and predicting the return time for such users is much easier. In order to investigate this further, we perform a pilot study in which we hold out a small sample of 1000 return time observations selectively chosen to be longer than 30 days from the proprietary dataset. The performance of different versions of the Cox model trained using the various schemes for handling recurrent observations discussed earlier is then tested at predicting these longer return time observations. The RMSE results are reported in table 8

	Test data of long return times				
	Weighted	Un-weighted	First Event	Last Event	Single day
RMSE	32.25	40.70	32.34	32.14	41.81

Table 8: RMSE for Long Return Time Prediction for different versions of the Cox Model on the Proprietary Dataset.

These results further show that both the un-weighted scheme and choosing observations from a single day, perform poorly at predicting longer return times. Since the focus of our methods is to find users which are not likely to return soon, these approaches may not be suitable for our application. Furthermore, the weighted scheme offers a good trade-off between using just the first events or just the last events per user in our model making it more suitable for our problem.

7. CONCLUSION

In this work, we have focused on the return time performance metric for free web services. We suggest that retention solutions driven by the projected return time of users can directly address the heart of the problem for web services, which is to encourage their users to frequently engage with their service. To facilitate such efforts, we formulate the problem of user return time prediction and define several covariates relevant to the problem. The Cox's proportional hazard model is proposed as the model of choice for this prediction problem due to several reasons including the ability to handle dynamics in user return rate with time and to incorporate the LOA information. A plot of the prediction performance scores against the LOA values allows a service to identify the right amount of gap since the user's last visit needed to start retention efforts. The performance of the hazard based model is found to surpass all the state-of-the-art baselines considered by us. Finally, we find that the ability of the Cox model to quantify the impact of several important covariates, including those related to user usage patterns, on user return rates to provide important insights that can guide future decision making for the service.

The Cox's model can further accommodate several complexities of the real-world quite well. For example, our analysis till now has been limited to static covariates. However, time-varying covariates including those pertaining to external factors such as holiday and weekend effects can be important for return time prediction and can be easily incorporated in the Cox Model [27]. In our final model for the large scale music service, we incorporated the effect of the day of the month covariate (Fig. 5) on the user return rates. Another direction for future research is to account for heterogeneities among users. Several solutions exist for either controlling for such differences between users [20] or for extracting different users segments through clustering [19] can also be applied to the return time prediction problem.

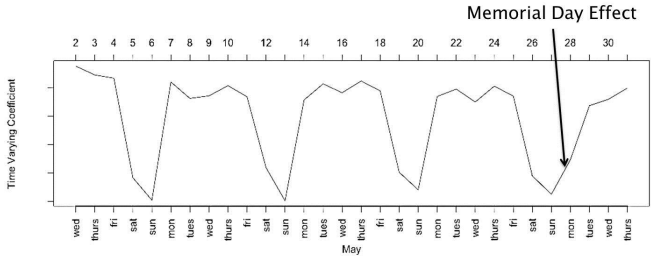


Figure 5: The regression coefficient for time-varying covariates corresponding to the different days of the month. The absolute values are omitted here.

8. REFERENCES

[1] http://www.last.fm.

[2] Simon Attfield, Gabriella Kazai, Mounia Lalmas, and Benjamin Piwowarski. Towards a science of user engagement (position paper). In *WSDM Workshop on User Modelling for Web Applications*, 2011.

[3] Alex Berson, Stephen Smith, and Kurt Thearling. *Building data mining applications for CRM*. McGraw-Hill New York, 2000.

[4] Wouter Buckinx and Dirk Van den Poel. Customer base analysis: partial defection of behaviourally loyal clients in a non-contractual fmcg retail setting. *European Journal of Operational Research*, 164(1):252–268, 2005.

[5] Jonathan Burez and Dirk Van den Poel. Handling class imbalance in customer churn prediction. *Expert Systems with Applications*, 36(3):4626–4636, 2009.

[6] O. Celma. *Music Recommendation and Discovery - The Long Tail, Long Fail, and Long Play in the Digital Music Space*. Springer, 2010.

[7] Kristof Coussement and Dirk Van den Poel. Churn prediction in subscription services: An application of support vector machines while comparing two parameter-selection techniques. *Expert systems with applications*, 34(1):313–327, 2008.

[8] David R Cox. Regression models and life-tables. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 187–220, 1972.

[9] Gideon Dror, Dan Pelleg, Oleg Rokhlenko, and Idan Szpektor. Churn prediction in new users of yahoo! answers. In *Proceedings of the 21st international conference companion on World Wide Web*, pages 829–834. ACM, 2012.

[10] Georges Dupret and Mounia Lalmas. Absence time and user engagement: evaluating ranking functions. In *Proceedings of the sixth ACM international conference on Web search and data mining*, pages 173–182. ACM, 2013.

[11] Peter S Fader, Bruce GS Hardie, and Ka Lok Lee. Rfm and clv: Using iso-value curves for customer base analysis. *Journal of Marketing Research*, pages 415–430, 2005.

[12] Rishab Aiyer Ghosh. Cooking pot markets: an economic model for the trade in free goods and services on the internet. *First Monday*, 3(2), 1998.

[13] Bing Quan Huang, M Tahar Kechadi, and Brian Buckley. Customer churn prediction for broadband internet services. In *Data Warehousing and Knowledge Discovery*, pages 229–243. Springer, 2009.

[14] Zainab Jamal and Randolph E Bucklin. Improving the diagnosis and prediction of customer churn: A heterogeneous hazard modeling approach. *Journal of Interactive Marketing*, 20(3):16–29, 2006.

[15] Michael D Kaplowitz, Timothy D Hadlock, and Ralph Levine. A comparison of web and mail survey response rates. *Public opinion quarterly*, 68(1):94–101, 2004.

[16] Marcel Karnstedt, Tara Hennessy, Jeffrey Chan, and Conor Hayes. Churn in social networks: A discussion boards case study. In *Social Computing (SocialCom), 2010 IEEE Second International Conference on*, pages 233–240. IEEE, 2010.

[17] Janette Lehmann, Mounia Lalmas, Elad Yom-Tov, and Georges Dupret. Models of user engagement. In *User Modeling, Adaptation, and Personalization*, pages 164–175. Springer, 2012.

[18] Han Liu, Atif Nazir, Jinoo Joung, and Chen-Nee Chuah. Modeling/predicting the evolution trend of osn-based applications. In *Proceedings of the 22nd international conference on World Wide Web*, pages 771–780. International World Wide Web Conferences Steering Committee, 2013.

[19] Patrick Mair and Marcus Hudec. Analysis of dwell times in web usage mining. In *Data Analysis, Machine Learning and Applications*, pages 593–600. Springer, 2008.

[20] CA McGilchrist and CW Aisbett. Regression with frailty in survival analysis. *Biometrics*, pages 461–466, 1991.

[21] Scott A Neslin, Sunil Gupta, Wagner Kamakura, Junxiang Lu, and Charlotte H Mason. Defection detection: Measuring and understanding the predictive accuracy of customer churn models. *Journal of Marketing Research*, pages 204–211, 2006.

[22] Joe Peppard. Customer relationship management (crm) in financial services. *European Management Journal*, 18(3):312–327, 2000.

[23] Jeffrey F Rayport. The truth about internet business models. *Strategy and Business*, pages 5–7, 1999.

[24] Werner J Reinartz and Vijay Kumar. On the profitability of long-life customers in a noncontractual setting: An empirical investigation and implications for marketing. *The Journal of Marketing*, pages 17–35, 2000.

[25] Bruno Ribeiro. Modeling and predicting the growth and death of membership-based websites. *Proceedings of the 23rd international conference on World Wide Web*, 2014.

[26] David C Schmittlein and Vijay Mahajan. Maximum likelihood estimation for an innovation diffusion model of new product acceptance. *Marketing science*, 1(1):57–78, 1982.

[27] Terry Therneau and Cindy Crowson. Using time dependent covariates and time dependent coefficients in the cox model. *The Survival Package (R help guide)*, 2013.

[28] Chih-Fong Tsai and Yu-Hsin Lu. Customer churn prediction by hybrid neural networks. *Expert Systems with Applications*, 36(10):12547–12553, 2009.

[29] Chih-Ping Wei, I Chiu, et al. Turning telecommunications call details to churn prediction: a data mining approach. *Expert systems with applications*, 23(2):103–112, 2002.

[30] Yaya Xie, Xiu Li, EWT Ngai, and Weiyun Ying. Customer churn prediction using improved balanced random forests. *Expert Systems with Applications*, 36(3):5445–5449, 2009.

[31] Jiang Yang, Xiao Wei, Mark S Ackerman, and Lada A Adamic. Activity lifespan: An analysis of user survival patterns in online knowledge sharing communities. In *ICWSM*, 2010.

[32] Elad Yom-Tov, Mounia Lalmas, Ricardo Baeza-Yates, Georges Dupret, Janette Lehmann, and Pinar Donmez. Measuring inter-site engagement. In *Big Data, 2013 IEEE International Conference on*, pages 228–236. IEEE, 2013.

[33] Yin Zhu, Erheng Zhong, Sinno Jialin Pan, Xiao Wang, Minzhe Zhou, and Qiang Yang. Predicting user activity level in social networks. In *Proceedings of the 22nd ACM international conference on Conference on information & knowledge management*, pages 159–168. ACM, 2013.