A Bayesian Point Process Model for User Return Time Prediction in Recommendation Systems

Sherin Thomas

A Thesis Submitted to

Indian Institute of Technology Hyderabad

In Partial Fulfillment of the Requirements for

The Degree of Master of Technology

भारतीय प्रौद्योगिकी संस्थान हैदराबाद

Indian Institute of Technology Hyderabad

Department of Computer Science and Engineering

June 2018

Declaration

I declare that this written submission represents my ideas in my own words, and where ideas or words of others have been included, I have adequately cited and referenced the original sources. I also declare that I have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in my submission. I understand that any violation of the above will be a cause for disciplinary action by the Institute and can also evoke penal action from the sources that have thus not been properly cited, or from whom proper permission has not been taken when needed.

(Signature)

(Sherin Thomas)

CS19MTECH11016

(Roll No.)

Approval Sheet

This Thesis entitled A Bayesian Point Process Model for User Return Time Prediction in Recommendation Systems by Sherin Thomas is approved for the degree of Master of Technology from IIT Hyderabad

Maunendra

(Maunendra Sankar Desarkar) Examiner
Department of Computer Science and Engineering
Indian Institute of Technology Hyderabad

(Manish Singh) Examiner
Department of Computer Science and Engineering
Indian Institute of Technology Hyderabad

(Dr. Srijith P.K.) Adviser
Department of Computer Science and Engineering
Indian Institute of Technology Hyderabad

V.B.Vineeth

(Dr Vineeth N Balasubramanian) Chairman
Department of Computer Science and Engineering
Indian Institute of Technology Hyderabad

Acknowledgements

I would like to thank my guide, Dr. Srijith P.K. for his support and valuable suggestion. His constant guidance, supervision, and invaluable knowledge were very helpful. I would also like to express my sincere gratitude to the Computer Science and Engineering department at Indian Institute of Technology Hyderabad for providing the inspiration and resources to help me complete my work successfully. I am thankful to my seniors and friends for their cooperation, support and words of encouragement in due course of my research. I humbly extend my gratitude to everyone who had helped me with this work over the years. Last but not the least, I am grateful to my family for the motivation and support.

Abstract

In order to sustain the user-base for a web service, it is important to know the return time of a user to the service. In this work, we propose a point process model which captures the temporal dynamics of the user activities associated with a web service. The time at which the user returns to the service is predicted, given a set of historical data. We propose to use a Bayesian non-parametric model, log Gaussian Cox process (LGCP), which allows the latent intensity function generating the return times to be learnt non-parametrically from the data. It also allows us to encode prior domain knowledge such as periodicity in users return time using Gaussian process kernels. Further, we capture the similarities among the users in their return time by using a multi-task learning approach in the LGCP framework. We compare the performance of LGCP with different kernels on a real-world *last.fm* data and show their superior performance over standard radial basis function kernel and baseline models. We also found LGCP with multitask learning kernel to provide an improved predictive performance by capturing the user similarity.

KEYWORDS: Return time prediction, Log Gaussian Cox Process, Poisson Process, Recommendation System, Multi-task learning, Multi-view learning.

Contents

Chapter 1

Introduction

1.1 Overview

Personalized recommendation systems provide custom-made recommendations to a user based on his interests and behaviour. Modeling the temporal dynamics of users in a recommendation system is invaluable to engage and retain users by providing recommendations that matches with the user interest. It provides useful information about the evolution of user interest. The available log of user histories can be exploited to tailor the services for each users as per the user's interests and behaviour. This has applications like recommending the right thing at the right time, market-basket analysis, advertisements, modeling drift in the trend etc. Appropriate recommendations and advertisements for the user at the right time are the keys for engaging the user in the service in a satisfactory way. Predicting the return rate of users is a prerequisite for this task.

The temporal pattern for each user varies in most of the cases. It can be captured by analyzing the interactions of the users with the system during use. Figure 1.1 illustrates a pattern of music listening activity of a user. The listening pattern varies across users and across the times. There are regions where the intensity of events are low and high. It is observed that if the frequency of interaction with an item decays with time, there are chances that the item will not be attractive further to users. Also the user characteristics influences the further interactions of users. Some users prefer to stick to a certain set of items whereas another set of user go with the current trends. Such latent behaviours of users can be exploited to provide recommendations to users according to their tastes. Learning the hidden pattern from the user's choices helps the service providers to suggest the most relevant items to the user when the available information base is huge. It also helps the users to find the chunk of data of their interest. The effort or interventions of user has to be reduced to provide a better experience in using online services.

In this thesis, the aim is to solve the problem of modeling the temporal dynamics of user activities. We focus on the task of predicting the return time of a user to begin a new music listening session. In particular, we model the return time of a user to a music web service called *last.fm.* We predict the next session start time for each user, given the historical data related to each users, using log-Gaussian Cox processes.We propose to use a Bayesian non-parametric point process which provides statistical procedures to model return time of users. It models the return time of users to be characterized by a latent stochastic intensity function. Specifically, we use a doubly stochastic

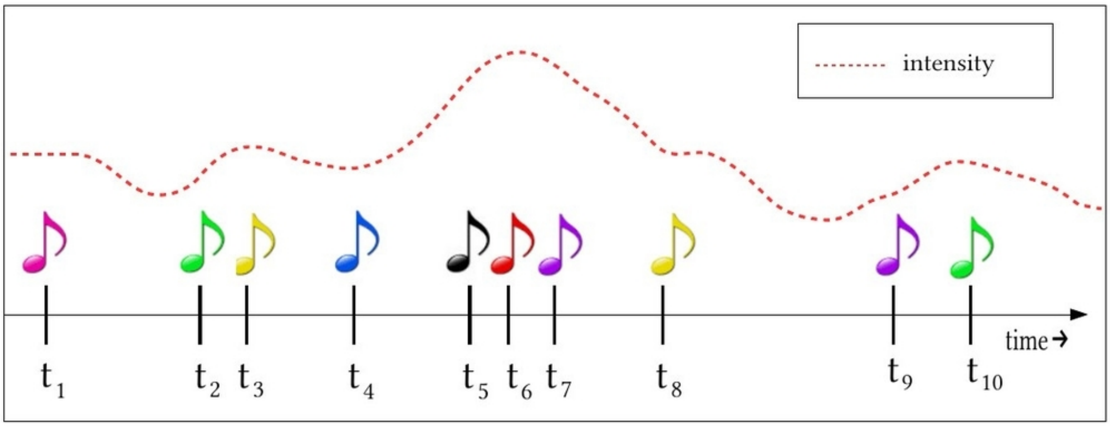


Figure 1.1: An illustrative example modeling the occurrences of music listening events at varying density.

inhomogeneous Poisson process, known as the Log Gaussian Cox Process[1]. It can learn the complex underlying intensity function non-parametrically from the data by assuming it to be coming from a Gaussian Processes (GP) [2] prior. We also capture the similarities across the users in this framework using a multi-task learning approach based on Gaussian processes [3]. The implicit user features are also considered by using a multi-view model. We compare the performance of the proposed approaches with various kernels against several baselines and demonstrate their usefulness on *last.fm* data.

1.2 Motivation

The traditional recommendation systems based on content-based recommendation, collaborative filtering recommendations etc. provides a qualitative representation of how users and items are related. But the strength of these relations also plays a vital role when it comes to making predictions about further interaction patterns of users with those items. Modeling the return times of user is a complex task as each users exhibits different return patterns at different points of time. The underlying pattern of return times of users can be captured using some functional forms. Using probabilistic frameworks for this task is beneficial as it can represent the complex relationship between users and items with time in a qualitative as well as in a quantitative manner, by means of a set of probability distributions. The uncertainties arising from the lack of enough information or the noise in the data can be represented by such a model and leads to an improved version of recommendation system. In the thesis, we evaluated the usefulness of probabilistic framework on the process of return time predictions of users to a service. We evaluated the performance of these methods by performing experiments on benchmark datasets released by Last.fm website.

1.3 Related Work

Personalized recommendation is one of the major topic of interest to researchers due to the availability of huge amount of data, which in effect limits the ease of interaction of a user with a service. In order to provide a custom-made recommendation list to a user, the user interests and behaviour has to be learned efficiently. Most of the works which aims to model this, considers a static behaviour of user for a period of time. The user return rates has to be modeled efficiently for appropriate and timely recommendations. In cases where the data is sparse, the ability to model the users degrades due to the lack of information about user behaviour.

The problem of modeling the return time of users was recently modelled using a deep learning framework, where recurrent neural networks were equipped with survival loss function [4] and data augmentation methods to capture shift in input data distribution [5]. Another method [5], was to solve the the session based recommendation task using a recurrent neural network with data augmentation and methods to capture the shift in input data distribution. Also, methods which builds a connection between recurrent neural networks and point processes[6] was also proposed. Point processes were used before to predict the return time of users, where a self exciting point process is combined with a low rank model to capture the temporal patterns in user activity [7]. The consumption behaviour of users with time was modeled using a hidden semi-Markov model (HSMM) in [8], which considered the latent features of users. A hazard based approach based on Cox proportional Hazard model [9] considered predicting the return time of users by considering various covariates like active weeks, visit number etc. The advantage of the proposed model, LGCP over these methods is that we don't have to provide the functional form of the intensity function as it is learnt non-parametrically from the data. Moreover, unlike the amount of data needed to train neural network models, it could generalize well from small data. We could also encode our prior knowledge on user behaviour such as periodicity and other patterns through the GP kernel which makes them more interpretable.

1.4 Publication

Part of this work has been accepted for publishing in the following conference proceedings.

Sherin Thomas, P. K. Srijith, and Michal Lukasik. 2018. A Bayesian Point Process Model for User Return Time Prediction in Recommendation Systems. In UMAP 18: 26th Conference on User Modeling, Adaptation and Personalization,July 811, 2018, Singapore, Singapore. ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3209219.3209261

Chapter 2

Probabilistic Models for Temporal Modeling

In order to model the hidden the pattern of user return times, we use the probabilistic frameworks like Gaussian process and Point process. In this chapter, we discuss these models which is used in our problem to learn the latent pattern for each user activities.

2.1 Gaussian Process

Gaussian process(GP) is a Bayesian non-parametric framework, which is a Gaussian probability distribution in a generalized form. Gaussian process can provide alternate approaches to solve the problems of regression, classification etc. GP is a non-parametric approach in that it provides a distribution over all possible functions that can represent the given data points. It is a Bayesian approach as it starts with a prior distribution over all functions and then come up with a posterior distribution of functions when new data points are observed.

Gaussian process has various properties that makes it more interesting. Since it is a non-parametric approach, the exact functional form for representing the data need not be fixed in prior, which provides the flexibility in choosing the functions. Also, in order to select the hyper parameters, extensive cross validations are not required, but they can be learnt by maximizing the marginal likelihood as in other Bayesian approaches. Gaussian process also helps in avoiding over-fitting as it considers a predictive distribution over the functions and averages it. It also encodes the uncertainties in the prediction by means of these distributions.

Consider a set of inputs $X = \{x_1, ..., x_n\}$ and the corresponding set of outputs $y = \{y_1, ..., y_n\}$ as in a regression problem, where the outputs are real scalar values. GP specifies prior over the functions which are objects of infinite dimensions and represents the relationship between the input X and output Y respectively. This prior information can be converted into a posterior when new data points are observed. We write the Gaussian process as,

$$f(x) \sim GP(m(x), k(x, x')) \tag{2.1}$$

where $m(x)$ is the mean function and $k(x, x')$ is the covariance function which is a positive definite kernel. The mean function specifies the expected value of the function. It is set to 0 in case of absence of any prior knowledge. The covariance function defines how the output covary as a function of input. If two inputs are deemed by the kernel to be similar, then the corresponding outputs of the functions at those points are also expected to be similar. A GP prior states that the joint distribution of function outputs $\{f(x_1), ..., f(x_n)\}$ corresponding to any finite set of inputs $\{x_1, ..., x_n\}$ is a multi-variate Normal distribution defined by the mean function $m(x)$ and the kernel function $k(x, x')$'

Let $X = \{x_1, ..., x_n\}$ and $f = \{f(x_1), ..., f(x_n)\}$, then the distribution is defined as follows.

$$p(f|X) \sim \mathcal{N}(\mu, \Sigma) \tag{2.2}$$

where $\mu = m(X) = \begin{bmatrix} m(x_1) \\ ... \\ m(x_n) \end{bmatrix}$ and $\Sigma = \begin{bmatrix} k(x_1, x_1) & ... & k(x_1, x_n) \\ ... & ... & ... \\ k(x_n, x_1) & ... & k(x_n, x_n) \end{bmatrix}$.

The posterior is defined as the joint probability over the observed (denoted as f) and unobserved outcomes (denoted as f_*) as,

$$p(f, f_*|X, X_*) \sim \mathcal{N}\left(\begin{bmatrix} \mu \\ \mu_* \end{bmatrix}, \begin{bmatrix} K(X, X) & K(X, X_*)^T \\ K(X, X_*) & K(X_*, X_*) \end{bmatrix} \right) \tag{2.3}$$

$K(X, X)$ is a matrix obtained by applying the kernel function to the observed values. It gives the similarities of the observed data points. Similarly, $K(X, X_*)$ gives the similarities of the observed and unobserved values that we are trying to obtain. $K(X_*, X_*)$ finds the similarities of the unobserved data points with each other.

2.1.1 Kernels

Kernel functions denote the similarity between the points. It represents the data in a higher dimensional space. The kernel $k(x_i, x_j) =< \phi(x_i), \phi(x_j) >$ takes the inner product of the points in the higher dimensional space. The kernels are constrained to be positive semi-definite for the covariance function. Some popular kernels are defined as follows:

Radial Basis Function(RBF) kernel

It is a useful when not much prior information about the data is available. It is also known as Squared Exponential kernel or Gaussian kernel. It is defined as,

$$k_{RBF}(x_i, x_j) = \sigma^2 exp\left(\frac{(x_i - x_j)^2}{2l^2} \right) \tag{2.4}$$

where l is the length scale, which defines the smoothness of the function. σ is the variance of the function. which defines how much the function values can vary from the mean.

Rational Quadratic(RQ) kernel

This kernel is similar to the one obtained by adding together many RBF kernels with varying length scales. It has the form,

$$k_{RQ}(x_i, x_j) = \sigma^2 \left(1 + \frac{(x_i - x_j)^2}{2\alpha l^2}\right)^{-\alpha} \tag{2.5}$$

where σ is the variance and l is the length scale of the function. α is the parameter that control the mixing of various length scales.

Periodic Exponential kernel

This kernel helps to capture the periodicity existing in the functional form of the data points. It is defined as,

$$k_{Periodic}(x_i, x_j) = \sigma^2 exp\left(- \frac{2sin^2(\pi|x_i - x_j|/p}{l^2}\right) \tag{2.6}$$

where l is the length scale and p is the period of the function which represents the distance between the repetitions in the function.

Multi-view kernel

There are different ways for combining these standard kernels in order to obtain a multi-view kernel. Two different kernels can be combined by multiplication or addition. Multiplying kernels helps in obtaining different high level properties that is otherwise not possible by using the kernels individually. Addition helps in modeling strong assumptions about the individual components that results in the sum, when there are different possible contexts. Additive kernels helps in extrapolation to points away from the training data.

Multi-task kernel

In order to model a multiple output GP and capture the correlation between them, a multi-task learning approach[3] can be utilized, where the GP kernel is parameterized by a matrix which represents the similarities between pairs of tasks.

$$k_{MULTITASK}(x_i, m_i),(x_j, m_j) = k(x_i, x_j)B_{m_i,m_j} \tag{2.7}$$

where B_{m_i,m_j} is a symmetric and positive semi-definite matrix capturing the similarities between the tasks. If the features of different tasks are available, then we can define the kernel as the product of different kernels.

2.1.2 Posterior Estimation

Given a GP prior $p(f|X)$ and a likelihood of observing the output values y given the latent function f, the GP posterior distribution of the functions that represent the data can be obtained using the Bayes theorem as follows.

$$p(f|y, X) = \frac{p(y|f)p(f)}{p(y|X)} \tag{2.8}$$

The predictive distribution over the latent function for the test data point x^* is then obtained using the approximated posterior.

$$p(f^*|X, y, x^*) = \int p(f^*|X, x^*, f)p(f|y, X)df \tag{2.9}$$

which can be used to find the predictive distribution over the test output for the given input data points.

$$p(y^*|X, y, x^*) = \int p(y^*|f^*)p(f^*|X, y, x^*)df^* \tag{2.10}$$

In case the likelihood and prior are non-conjugate, the posterior can be approximated using Laplace approximation, where the posterior $p(f|y, X)$ is approximated by a Gaussian distribution $q(f|y, X)$ based on the first and the second derivative of the logarithm of the unnormalized posterior [2].

2.2 Point Process

A Point process is a mathematical framework to represent a collection of random points located in some mathematical space. Point processes has various applications in real time such as counting problems, population recording processes, plotting occurrences of natural events like earthquakes, floods etc. It can also be used to predict the number of points occurring in a given interval or to predict the time at which the next event occurs, given some history of events. Point process models points in the space using an intensity function $\lambda(t)$. Higher value for this intensity function implies higher density of points and vice versa. There are different types of Point processes like Hawkes process and Poisson process.

2.2.1 Poisson Process

This is a specialized form of Point process which is used for modeling counting problems. As a simple definition, if we are counting the number of events in a sub region of space, such that the events is different sub regions are independent and Poisson distributed, then the process obtained is known as a Poisson Point process.

There are two types of Poisson process namely, homogeneous Poisson process and inhomogeneous Poisson process. If we consider the case of a temporal space and if the intensity function of a Poisson process is constant with time i.e, $\lambda(t) = \lambda$, then such a process is known as homogeneous Poisson process. In case of an inhomogeneous Poisson process, the event occurs at a variable rate which is a function of the space we consider. If the number of events occurring in an interval $[s, e]$ is denoted as y, it is Poisson distributed with the variable rate parameter $\int_s^e \lambda(t)dt$ as,

$$P(y|\lambda(t), [s, e]) = Poisson\bigg(y|\int_s^e \lambda(t)dt\bigg) \tag{2.11}$$

$$= \frac{(\int_s^e \lambda(t)dt)^y exp(-\int_s^e \lambda(t)dt)}{y!}$$

Cox Process

Cox Process is a special case of an inhomogeneous Poisson Point process. In an inhomogeneous Poisson process the points are drawn from a stochastic process with a latent intensity function. Whereas a Cox process is a doubly stochastic inhomogeneous Poisson process as the latent intensity function is also drawn a stochastic process.

Log Gaussian Cox Process

Log Gaussian Cox Process(LGCP) is a Cox Process in which the latent intensity function $\lambda(t)$ is modeled using a function $f(t)$ drawn from Gaussian Process as follows.

$$\lambda(t) = exp(f(t)) \tag{2.12}$$

Taking the exponential ensures that the intensity function is positive.

Survival Model

Here we discuss the survival model aspect of Poisson process. Consider two random variables, N_t is a discrete random variable representing the number of event in a time interval $(0, t)$ and T is a continuous random variable which represents the time till the first event occurs. Then the survival model represents how likely it is, that the next event happens only after time t and it is defined as the probability that no event happens till T. Both the variables are related to each other as follows.

$$S(t) = P[T > t] = P[N_t = 0] = exp\bigg(- \int_0^t \lambda(s)ds \bigg) \tag{2.13}$$

If $p(t)$ represents the probability density function of T, then $p(t)$ is related to the intensity function $\lambda(t)$ and survival function $S(t)$ as,

$$p(t) = \lambda(t)S(t) = \lambda(t)exp\bigg(- \int_0^t \lambda(s)ds \bigg) \tag{2.14}$$

which gives the probability of occurrence of an event at time t.

Chapter 3

Modeling Temporal Dynamics in Recommendation Systems

In this chapter, we consider the problem of modeling the temporal dynamics of user activities in a recommendation system, in order to predict the time at which the user returns to the service again. We model the temporal frequencies of the user activities by considering the histories of user activities and the demographic information of users. We capture the similarities across users by considering a multi-task learning approach. Our model is based on Poisson Point process, which can efficiently model count data in a continuous space using an underlying intensity function of a Poisson distribution. Specifically, we use the Log Gaussian Cox Process model which is a Poisson Point process in which the logarithm of the intensity function is obtained from a Gaussian Process prior.

To summarize, the main contributions in this chapter are as follows.

- Propose to use Bayesian non-parametric model, log Gaussian Cox Process to model return time prediction of users.
- Demonstrating how multi-task learning can model similarities in the session activity pattern of users and improve performance.
- Incorporating user demographic features into the model and showing the improvement in the results.
- Running comparison study of various kernels in LGCP, in how they capture user activity. We find that considering the kernel combination leads to superior performance over standard radial basis function kernel.

3.1 Problem Definition

In this work, we aim to solve the problem of modeling the temporal dynamics of user activities following the work in [4]. We focus on the task of predicting the return time of a user to begin a new music listening session. Consider a time interval of $[0, T]$ where T is the time point up to which we have the log of user session start times. Let us consider a data with M users with each user

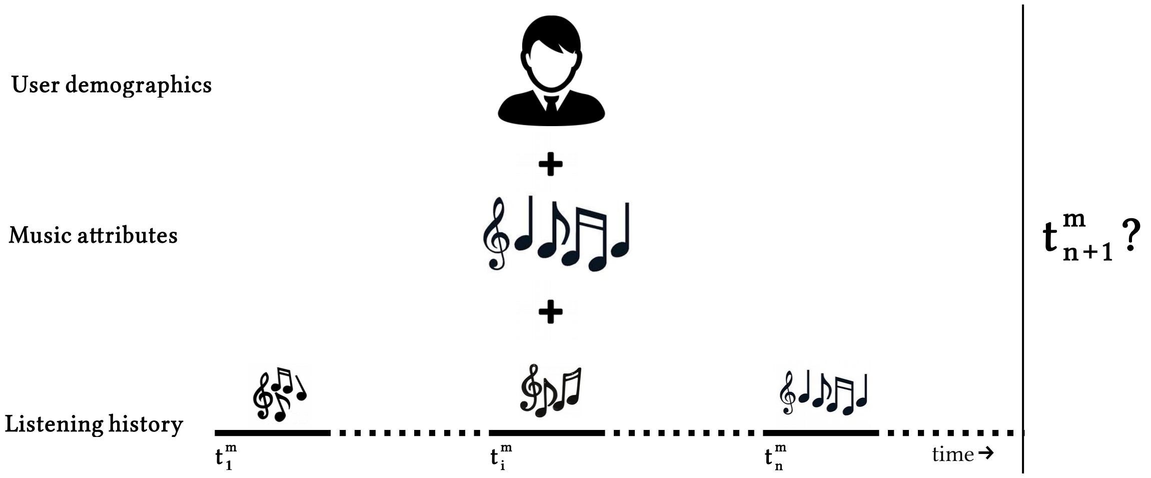


Figure 3.1: Modeling user return time of a user. Each user event is specified by the start time t_j^m of a session of activities. Given a user log of activities of n sessions along with the user and item(music) features, the model predicts the next session start time, t_{n+1}^m.

being associated with a set of sessions. Let $\mathbf{t}_m = \{t_n^m\}_{n=1}^{N_m}$ denote the sessions associated with user m with t_n^m denoting the start time of the n^{th} session and N_m is the number of sessions for user m. We also have the demographic information associated with each users including the age, gender, location and registered time. The item information includes all the music listened by the user in each sessions. We predict the next session start time for each user as shown in Fig. 3.1, given the historical data related to each users along with the user and item features, using log-Gaussian Cox processes(see Section 2.2.1). We take the data for 3 months to train the model and then predict the return times of users in the succeeding one month.

3.2 Model

The users interest to start a new session and its duration changes over time. This can be captured using an inhomogeneous Poisson process (IPP) with a time varying intensity $\lambda(t)$ [10]. In an IPP model, the number of events occurring in an interval $[s, e]$ is Poisson distributed as in Eq. (2.11). Given the last event happened at time s, the probability of occurrence of event at time e is given as

$$p(e|\lambda(t), s) = \lambda(e)exp(-\int_s^e \lambda(t)dt) \tag{3.1}$$

Users exhibit complex temporal behavior and it is difficult to come up with an appropriate intensity function capturing their temporal behavior. This motivates us to use a doubly stochastic inhomogeneous Poisson process framework, log Gaussian Cox Process (LGCP), where the logarithm of the time varying intensity function is assumed to come from a Gaussian Process prior [1]. This allows us to learn the intensity function non-parametrically from the data in addition to specifying the domain knowledge through the GP kernel. The intensity function for a user m at a session starting time t_n^m is defined as,

$$\lambda^m(t_n^m) = exp(f^m(t_n^m)) \quad \text{where,} \quad f^m(t) \sim \mathcal{GP}(\mu^m(t), cov^m(t, t') \tag{3.2}$$

where μ^m is the mean function and cov^m is the covariance function of a GP for an user m. The covariance function specified through a positive definite kernel $k^m(t,t')$ determines various properties of the intensity function such as its periodicity, smoothness etc. Taking the exponential serves the purpose of ensuring the positivity of the intensity function.

Given the latent function f^m for a user m, the probability that the user will return at time t_n^m while his last session start time was t_{n-1}^m is obtained by combining (3.1) and (3.2),

$$p(t_n^m|f^m(t),t_{n-1}^m) = exp(f^m(t_n^m))exp(-\int_{t_{n-1}^m}^{t_n^m} exp(f^m(t))dt) \tag{3.3}$$

The likelihood of occurrence of all $\mathbf{t}_m$ sessions associated with an user m is given as,

$$\begin{aligned} L(t_1^m,t_2^m,...,t_n^m) &= \prod_{j=1}^{n} exp(f^m(t_j^m))exp(-\int_0^T exp(f^m(t))dt) \\ &= exp(-\int_0^T exp(f^m(t))dt + \sum_{j=1}^{n} f^m(t_j^m)) \end{aligned} \tag{3.4}$$

To eliminate the difficulties arising in computations due to integration, the likelihood in (3.4) is approximated by considering sub-intervals of T and assuming to have a constant intensity in those sub-intervals. Let the interval $[0,T]$ be divided into S sub-intervals, with each sub-interval s having a centre as t_s, l_s the length of the sub-interval and y_s^m denoting the number of sessions by an user m in the given sub-interval. Then, the approximated likelihood is given as,

$$\hat{L}(t_1^m,t_2^m,...,t_n^m) = p(\mathbf{y^m}|f^m) = \prod_{s=1}^{S} Poisson(y_s^m|l_s exp(f^m(t_s))) \tag{3.5}$$

A user specific Gaussian process prior $p(f^m)$ is used for the underlying latent function f^m. Using the Bayes Theorem,the posterior distribution over the latent function f^m can be obtained as,

$$p(f^m|\mathbf{y^m}) = \frac{p(\mathbf{y^m}|f^m)p(f^m)}{p(\mathbf{y^m})} \tag{3.6}$$

Since the likelihood is a Poisson distribution and the prior is a Gaussian, the posterior distribution is intractable. Hence, an approximate posterior $q(f^m)$ is obtained using Laplace approximation [2] which fits a Gaussian around the mode of the posterior.

The predictive distribution over the latent function for the test data point t_*^m is then obtained using the approximated posterior.

$$p(f^m(t_*^m)|\mathbf{y^m}) = \int p(f^m(t_*^m)|f^m(\mathbf{t^m}))q(f^m)df^m \tag{3.7}$$

Using the predictive distribution over latent function in (3.7), the intensity value at time t_*^m can be obtained

$$\lambda^m(t_*^m) = \int exp(f^m(t_*^m))p(f^m(t_*^m)|\mathbf{y^m})df^m(t_*^m) \tag{3.8}$$

The expected number of events y_*^m in an interval with length l_s, centred at t_*^m will be Poisson distributed with rate $l_s \lambda^m(t_*^m)$. The various model hyper-parameters such as the kernel parameters are learnt by maximizing the marginal likelihood $p(\mathbf{y^m})$.

Once we learn the intensity function from LGCP model, the exact return times of users are predicted by sampling time from a proposed exponential distribution using Ogata's thinning algorithm as outlined in Algorithm (1) [11].

Algorithm 1 Ogata's thinning algorithm

```
Input: Intensity function $\lambda(t)$ , last session start time $u$ , upper bound on time $T$,
Initialize $t = 0$ , $S = \{\}$
$\beta \leftarrow max(\lambda(t)) \; \forall t \in [u, T]$
Sample next arrival time $s$ from $exp(1/\beta)$
Generate random number $u \sim Uniform([0, 1])$
Set $t \leftarrow t + s$
if $u <= \frac{\lambda(t+s)}{\beta}$ then
    $S \leftarrow S \cup t$
end if
Return: $S$
```

3.2.1 Multi-task learning

We explore the similarities existing across different users in order to learn better intensity functions for a user. This is achieved by using a GP multi-task learning model [3] which learns the user similarity through the covariance function(see Section 2.1.1). In this approach the GP kernel is defined jointly over users and their return times. This can be obtained as a product of two kernels, one over the users and the other over return times (LGCP-Multitask). If the user demographics are available, then we take the product of the time kernel with the user feature kernel as

$$k_{MTL-A}((m,t),(m',t')) = k(t,t')k(m,m') \tag{3.9}$$

In the absence of user features, the user kernel is parameterized by a matrix which captures the similarities among the user activities and is learnt from the data. For two users m and m' and their corresponding return times t and t', the multi-task learning kernel is defined as

$$k_{MTL-B}((m,t),(m',t')) = k(t,t')\mathbf{B}_{m,m'} \tag{3.10}$$

Here, the matrix B captures the similarities across users. This is learnt by maximizing the marginal likelihood $p(\mathbf{y^m})$ of the model.

3.2.2 Multi-view learning

In a multi-view model, we take into account the inherent properties of items which again accelerates the performance of the model. In this approach the GP kernel is defined jointly over items and the user return times(see Section 2.1.1). This can be obtained as a sum of two kernels, one over the features of items and the other over return times (LGCP-Multiview). For two users m and m' with features $\mathbf{x}_m$ and $\mathbf{x}_{m'}$, and their corresponding return times t and t', the multi-view learning kernel is defined as

$$k_{MVL}((\mathbf{x}_m, t), (\mathbf{x}_{m'}, t')) = k(\mathbf{x}_m, \mathbf{x}_{m'}) + k(t, t') \tag{3.11}$$

Here the feature $\mathbf{x}_{m'}$ is taken as a combination of the features of music listened by the users in each sessions. Item features like music track name are encoded into vector representations and combined to form the feature vector.

Chapter 4

Experimental Evaluation

We evaluate the performance of the proposed approaches in predicting the return time of users to the service. The proposed models are compared with several other baseline models on a real world data set *last.fm* which consists of music listening times of users over some years.

4.1 Dataset

The publicly available *last.fm* [12] data collected from the *last.fm* API comprises of the music listening log of 992 unique users, with a total of 19,150,868 listening events spanning from 2004 to 2009. Each event is a tuple $< user, artist, song, timestamp >$ representing a listening event. Each user's profile consists of features like age, gender, country and sign-up timestamp.

4.2 Experimental Setup

The dataset is split into sessions by considering a time gap between consecutive listening events for a user. If two events are separated by a gap of 1 hour or more, then those consecutive events are assumed to belong to two different sessions [4]. This results in the dataset getting split into 741334 sessions in total for all users.

The user features (age,gender,country) are encoded as binary representations. The sign-up timestamp is split into intervals of 5 years and then encoded as binary representations. The artist name is considered as the item(music) feature, which is also taken in a binary encoded representation. For each sessions, the element wise sum of vector representations of all music listened by a user in that session is taken. The feature vector corresponding to each session is a concatenation of the user feature vectors and the summated item feature vector.

The pre-processed session data is split into training/testing set by taking 3 consecutive months of data for training and 1 month data for testing purpose. Users with less than 100 listening events in training set and 50 events in test test are considered inactive and hence removed. This results in 394 active users with 243 sessions on an average per user. Each user data for 4 months is split into equal bins of 24 hours each, resulting in 90 bins for training purpose and 30 bins for testing.

An LGCP model for each user is learnt from the training data as explained in Section 3.2 and is used to predict the user session start times on the test data. We use LGCP models with different kernels (see Section 2.1.1) including the Rational Quadratic, Periodic, and Radial Basis Function(RBF) [2]. We also use LGCP with multitask learning kernel which captures similarities in user activities.

4.3 Evaluation Metrics

In order to evaluate the model, we use mean absolute error(MAE) and root mean square error(RMSE) to evaluate the differences between the actual and the predicted return time values for each user. Since the data varies in size for each user, we take the micro average of the errors to obtain the final result.

4.4 Baselines

We use the following baseline models for comparison.

4.4.1 Homogeneous Poisson process (HPP)

Here the intensity is assumed to be same all through out the time period. The intensity value λ is estimated as the frequency of occurrences of all events in the given time period of training data(3 months). Inter-arrival times are sampled from an exponential distribution with rate parameter equal to λ.

4.4.2 Linear Regression (LR)

We learn a linear function which predicts the next session start time from the previous session start time.

$$t_{n+1} = b_0 + (b_1 \times t_n) \tag{4.1}$$

4.4.3 Gaussian Process Regression (GPR)

Gaussian Process Regression is also a non-parametric probabilistic model. We learn a non-linear function using a GP model with RBF kernel which predicts the next session start time from the history of session start times. If x_i is the history of session start times and y_i is the next session start time to be predicted, then,

$$y_i = f(x_i) + \epsilon_i \tag{4.2}$$

where $f \sim GP(0, K)$ and $\epsilon_i \sim \mathcal{N}(0, \sigma^2)$, i.e. the prior on f is a Gaussian Process and the likelihood is a Gaussian, therefore the posterior is tractable and hence it is also a Gaussian Process.

4.4.4 Recurrent neural network (RNN)

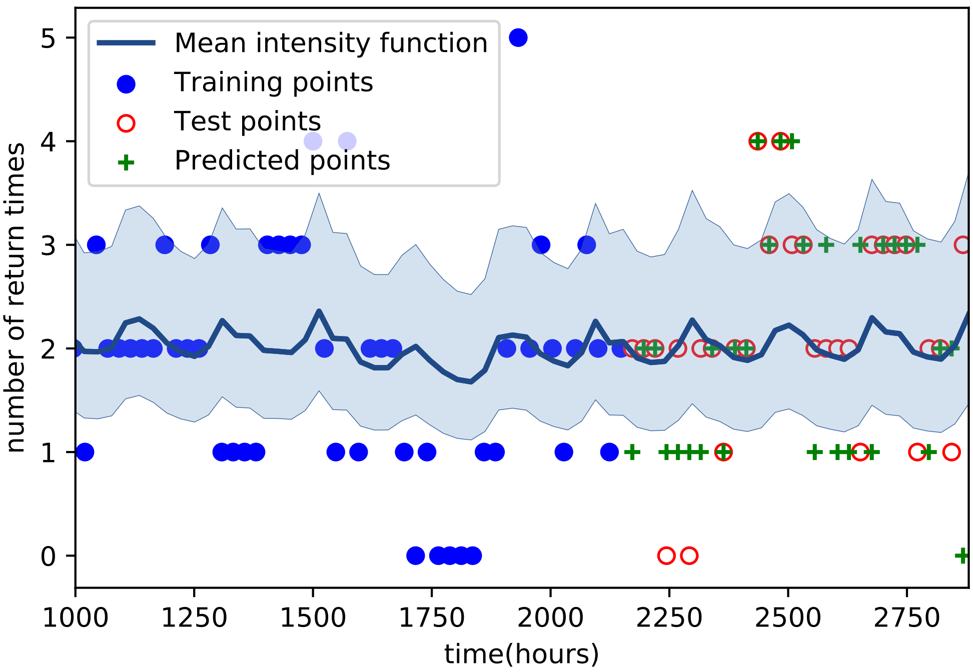
We learn a recurrent neural network specifically a Long Short-Term Memory (LSTM) with one hidden layer and 50 neurons which predicts the next session start time from the history of previous start times.

4.5 Results

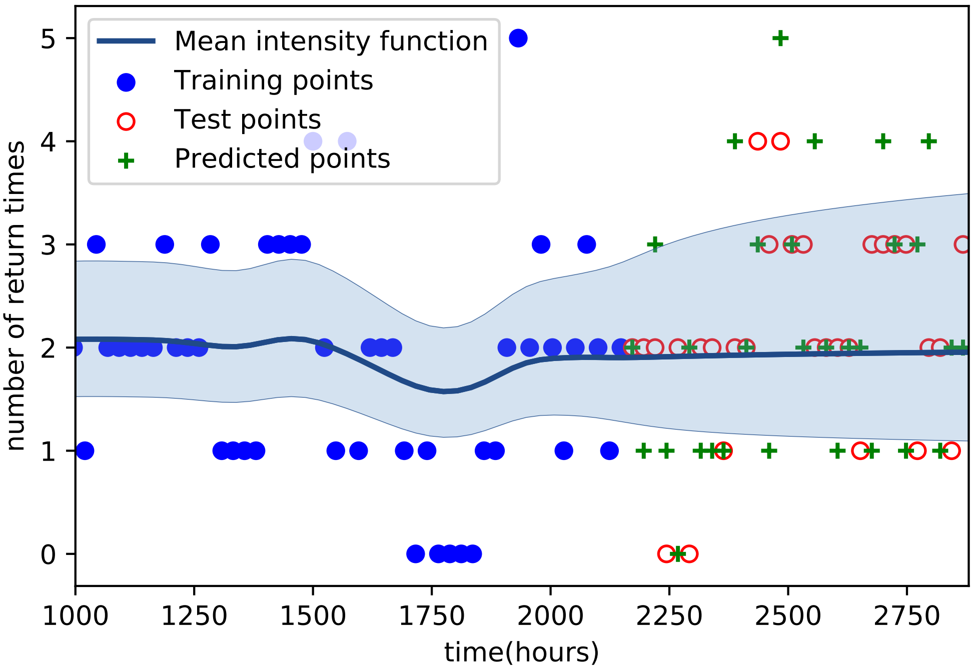
Method	Kernel	MAE	RMSE
LGCP	Periodic	9.37	19.45
	Rat Quad	8.68	18.74
	Periodic + Rat Quad	9.22	20.48
	RBF	15.89	22.44
LGCP-Multitask-A	Periodic	9.34	19.48
	Rat Quad	8.70	18.75
	Periodic + Rat Quad	9.08	19.71
	RBF	15.90	22.46
LGCP-Multitask-B	Periodic	8.89	19.25
	Rat Quad	8.69	18.76
	Periodic + Rat Quad	8.90	19.07
	RBF	15.87	22.41
LGCP-Multiview	Periodic	8.31	19.04
	Rat Quad	8.68	18.74
	Periodic + Rat Quad	8.52	19.04
	RBF	8.63	18.76
HPP		9.41	22.02
Linear Regression		10.25	22.56
GP Regression	RBF	10.30	22.98
RNN		11.05	20.46

Table 4.1: Mean Absolute Error(MAE) and Root Mean Squared Error(RMSE) between the actual and predicted user return time for proposed methods and baselines on the *last.fm* data.

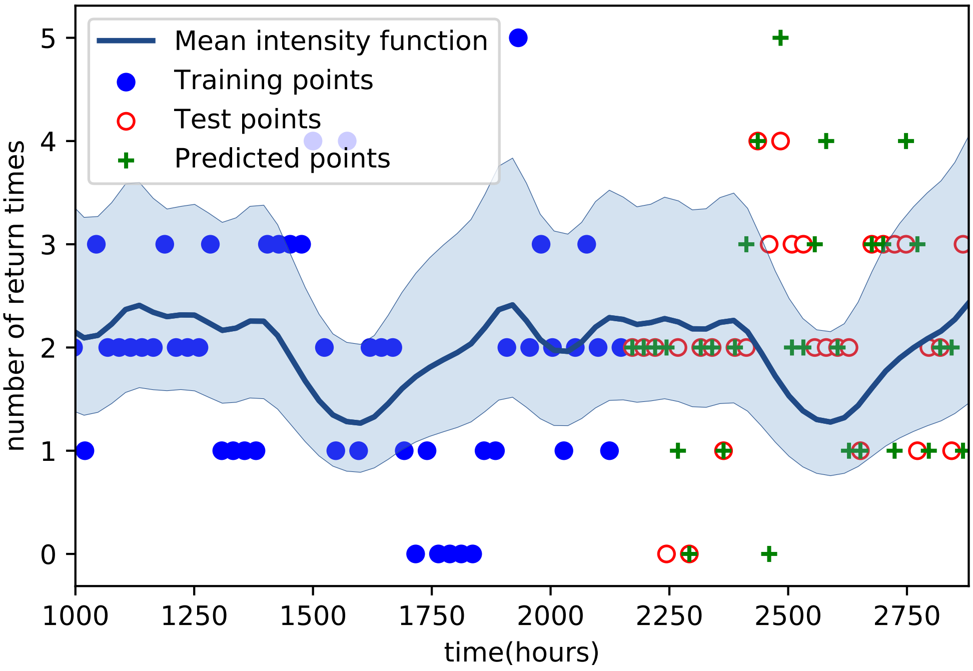
Table 4.1 compares the performance of various models in predicting the return time of users in terms of MAE and RMSE scores on the *last.fm* data. We obtain the predictive performance of LGCP, LGCP-Multitask-A,LGCP-Multitask-B and LGCP-Multiview approaches with various kernels over time such as Periodic, Rational Quadratic(RQ), RBF and a combination of Periodic and RQ. We find that the standard kernel used in GP models, RBF kernel, performs poorly in this data when the features are not considered. This is due to the complex temporal patterns exhibited by users for their session start times, which depends on the attributes of users and items. The RBF kernel typically models smoothly varying functions and is not suitable to model this situation. RQ kernel is obtained by considering a combination of RBF kernels with different length-scales and could model such complicated behaviour patterns better than RBF which uses a single length scale. This is evident from the experimental results where we found that RQ outperforms other kernels and baselines. The Periodic kernel could model the periodicity in the data (for instance, users tend to be more active on weekends) and are found to perform better than RBF but fails to capture other complex behavioral patterns captured by RQ. Combining the features with the Periodic kernel shows superior performance as it in effect captures both periodicity and feature dependency in the data. Combining RQ with Periodic shows a performance which lies midway between that of Periodic and RQ. All the LGCP models with these kernels (except RBF) outperformed the baseline approaches such as HPP, linear regression, GP regression, and RNN. LGCP-Multiview which considered user similarity through user features improved the performance for most of the kernels, while it remained similar to other cases for RQ kernel. However, LGCP-Multitask-A and LGCP-Multitask-B brought better improvements in performance with Periodic and $RQ + Periodic$ kernel while it retained the performance with other kernels. This corroborates the fact that considering other users with similar activity pattern could improve the predictive performance for an user. But considering similarity through user features is also effective. The LGCP-Multitask-A and LGCP-Multitask-B models shows performances which are comparable. Figure 4.0 plots the intensity functions for a user learnt by LGCP for the $RQ + Periodic$ kernel.



(a) LGCP.



(b) LGCP-MULTITASK-B.



(c) LGCP-MULTIVIEW.

Figure 4.0: Intensity function learnt for a user using LGCP, LGCP-MULTITASK-B and LGCP-MULTIVIEW models with $RQ + Periodic$ kernel on *last.fm* data. The x axis denotes the time and the y axis denotes the number of user returns within a $24h$ interval. The dark line denotes the predictive mean and the shaded region denotes the predictive variance. Note that we sample the predictions rather than using the predictive mean.

Chapter 5

Conclusion

We introduced a Bayesian non-parametric point process, log-Gaussian Cox process, to model the return time of users in recommendation systems. It learns the intensity function non-parametrically from the data and models the complex temporal behavioral patterns exhibited by users in their session start times. We also used a multi-task and multi-view learning approach within the LGCP framework to learn the intensity function for a user from users with similar activity pattern and also from the implict features of users and items. In multi-task model, we learnt the user similarity matrix from data whereas in multi-view model, we obtained it as a kernel over user and item features along with time kernel. This captures similarities across users based on features. The predictive performance of the proposed models were evaluated on the real world online music data *last.fm*. Various kernels were considered within the LGCP, LGCP-Multitask-A, LGCP-Multitask-B and LGCP-Multiview models to capture the user behaviour and we found them to perform better than the standard RBF kernel and other baselines. We also found that considering the user features through LGCP-Multiview could improve the predictive performance.

References

[1] J. Mller, A. R. Syversveen, and R. P. Waagepetersen. Log Gaussian Cox Processes. *Scandinavian Journal of Statistics* 25, (1998) 451–482.

[2] C. E. Rasmussen and C. K. I. Williams. Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning). The MIT Press, 2005.

[3] M. A. Álvarez, L. Rosasco, and N. D. Lawrence. Kernels for Vector-Valued Functions: A Review. *Found. Trends Mach. Learn.* 4, (2012) 195–266.

[4] H. Jing and A. J. Smola. Neural Survival Recommender. In WSDM. 2017 515–524.

[5] Y. K. Tan, X. Xu, and Y. Liu. Improved Recurrent Neural Networks for Session-based Recommendations. *CoRR* abs/1606.08117.

[6] N. Du, H. Dai, R. Trivedi, U. Upadhyay, M. Gomez-Rodriguez, and L. Song. Recurrent Marked Temporal Point Processes: Embedding Event History to Vector. In KDD. 2016 1555–1564.

[7] N. Du, Y. Wang, N. He, and L. Song. Time-sensitive Recommendation from Recurrent User Activities. In NIPS. 2015 3492–3500.

[8] K. Kapoor, K. Subbian, J. Srivastava, and P. Schrater. Just in Time Recommendations: Modeling the Dynamics of Boredom in Activity Streams. In WSDM. 2015 233–242.

[9] K. Kapoor, M. Sun, J. Srivastava, and T. Ye. A Hazard Based Approach to User Return Time Prediction. In KDD. 2014 1719–1728.

[10] S. Lee, J. R. Wilson, and M. M. Crawford. Modeling and Simulation of a Nonhomogeneous Poisson Process having Cyclic Behavior. *Communications in Statistics Simulation* 20, (1991) 777–809.

[11] Y. Ogata. On Lewis' simulation method for point processes. *IEEE Transactions on Information Theory* 27, (1981) 23–30.

[12] O. Celma. Music Recommendation and Discovery in the Long Tail. Springer, 2010.