ΟΙΚΟΝΟΜΙΚΟ
ΠΑΝΕΠΙΣΤΗΜΙΟ
ΑΘΗΝΩΝ

ATHENS UNIVERSITY
OF ECONOMICS
AND BUSINESS

School of Information Sciences and Technology
Department of Statistics
M.Sc. Program in Applied Statistics
Athens University of Economics and Business

MSc Dissertation

Clickstream analysis with timestamps

Author: Eleni Gkiouzepi
Supervisor: Prof. Dimitris Karlis

Department of Statistics

Athens, Greece
Date: November 2019

Abstract

Clickstream data contain important information about user's browsing behavior on a website. In this study, a Markov model, more specifically the Continuous Time Finite State Markov Chain (CTMC) is proposed to model the data. The pages in the clickstream data are categorized based on their content. These categories become the user-states in the model. The sequence of states for a particular visit by a user (session) becomes the chain. Using real data data and the CTMC model developed by Albert, the Q matrix is calculated (Albert et al. [1962]). This Q matrix is then used to calculate the probability of a user's next movement on the website. We show how path information can be categorized and modeled using a mixture of first order CTMC model. By using this model, we perform model based clustering and we group our data. The basic tool for this aggregation method is the EM algorithm.

Contents

List of Symbols **7**

1 Clickstream analysis **13**
1.1 Introduction 13
1.1.1 Motivation 13
1.1.2 Background 13
1.1.3 Purpose of this study 15

2 Markov Chains and Transition Functions **17**
2.1 Continuous-Time Markov Chains 17
2.1.1 Definition and the minimal construction of a Markov chain 17
2.1.2 Transition probabilities for finite state space 18
2.1.3 The infinitesimal generator matrix 19
2.1.4 The space of sample functions 20
2.2 Mixture Models 23
2.2.1 From Factor Analysis to Mixture Models 23
2.2.2 Identifiability 25
2.2.3 Probabilistic Clustering 26
2.2.4 Classification-semi supervised learning 26
2.2.5 Learning Mixture Models from Data 26
2.2.6 A Mixture of First Order Continuous Time Markov Models 27

3 EM algorithm **31**
3.1 Outline the EM algorithm for mixture models 31
3.2 The E-Step for Mixture Models 32
3.3 The M-Step for Mixture Models 32

3.4 EM algorithm for the Mixture Markov Model 33
3.5 Initialization and Convergence Issues for EM 34
3.6 Selecting the number of clusters 34

4 Data Description 35
4.1 Dataset analysis 37
4.2 Data preparation 38
4.3 Results 48
4.3.1 Finite mixture modelling with EM algorithm 48

5 Conclusion 51

Appendix A R Script 53

List of Symbols

$A_T^{(k)}(i)$	Length of time state i is observed in k observations
$F(x)$	The cummulative distribution function
a	The rate of transitions, the parameter in an exponential distribution
J	The number of states in the system
$N_T^{(k)}(i,j)$	Number of transitions from state i to state j in k observations
Q	The infinitesimal generator matrix for the CTMC
$P_{i,j}(t)$	The probability of jumping from state i to state j in time t
$f(x)$	The probability distribution function
$q_{i,j}$	The (i,j)th element of the Q matrix
$Z(t)$	The observable outcome of the system at time t
G	The number of groups

T_i	The total time spent in ith state
a_g	The initial probability vector

List of Figures

4.1 Frequency distribution of number of clicks per session in log scale at y-axis 40
4.2 Frequency distribution of number of clicks per item in log scale at y-axis 41
4.3 Frequency distribution of number of clicks per categories . . . 42
4.4 Most clicked categories 43
4.5 Complete cases: Frequency distribution of number of clicks per session in log scale at y-axis 44
4.6 Complete cases:Frequency distribution of number of clicks per item in log scale at y-axis 45
4.7 Complete cases:Frequency distribution of number of clicks per categories 46
4.8 Histogram of time spent (in seconds) on any path in log scale at y-axis 47
4.9 Histogram of time spent (in seconds) on any page in log scale at y-axis 47

List of Tables

4.1 Initial raw dataset *yoochoose-clicks.dat*. 38
4.2 Dataset filtered - only complete cases, with no missing values. 39
4.3 Data derived from 371984 calculated paths 42
4.4 BIC results for different number of groups. 49

Chapter 1

Clickstream analysis

1.1 Introduction

1.1.1 Motivation

Currently, online stores target visitors-users using many types of information,such as demographic characteristics, purchase history , and how the user find the site(e.g through a bookmark, search engine, or link on an email promotion). Another source of users information is the data which record the navigation path that a user takes through a Web site(Montgomery and Faloutsos [2001]).

On a Web site click stream analysis or clickstream analytics is the process of collecting, analysing and reporting aggregate data. So path data or clickstream data can inform us about the user's goals,interests and future purchases. More particularly clickstream data are very important for analysing user's behaviour and can be applied for example in online marketing and anti-terrorism. Moreover many platforms such as Face book and Google Ads rely on these data from what a user click's and from what he doesn't,so getting a better view about user's behaviour leads to improvements in these platforms.(e.g. advertising products, customer's experience).

1.1.2 Background

Brodwin et al. [1995] was from the first who referred to the clickstream analysis. According them, the website collects information about each web visitor and provides customized information that meets visitor's preferences. Afterwards Chatterjee [2003] by analyzing the clickstream data identified a group of users who were more likely to click a banner while also identifying practices that have no effect on the rate of click-through by the users.

Di Scala et al. [2004] analyzed clickstream data with means of Markov chains. The purpose of their research was to measure the effectiveness of a website. They used past clickstream data and a multi-level Markov model, to rank the pages and provide a measure of the effectiveness of the site. The use of Markov models to model web browsing patterns was first proposed by Pitkow and Pirolli [1999]. Their models treated each and every web page as unique states and didn't group them together.Today their model is no longer applied. Deshpande and Karypis [2004] suggested a modified Markov model. They began from a K-order Markov chain and started to reduce the number of states towards lower order Markov models. So their suggested model was as accurate as Pitkow and Pirolli's model but less complex. The main focus of their research was to determine the page that most likely the user would visit next. Montgomery et al. [2004] used a dynamic multinomial probit model. This model consists of a Vector Autoregressive component with a Hidden Markov chain in the background. They incorporated the idea of categorizing the pages into eight categories. Their model analyses the page-by-page viewings of a visitor as they browse through a web site . Scott and Hann [2006] in their working paper proposed a session-level Hidden Markov with a page-level Hidden Markov model. This model is an improvement over Montgomery's model but also increases the complexity of the model and cannot be implemented in real time. In their study, they chose to use discrete time units and suggested that continuous time units would probably produce better results. Lee et al. [2001] developed a model to visualize the clickstream data . The model allowed the website operator to better analyze the data to find trends in user's browsing behavior. Although the system developed uses live data, this technique is time consuming. Mobasher et al. [2001] used pattern recognition to group users based on their browsing behavior. Once the group was identified, the group's historical data were used to determine the user's probable action. Some of the data used to determine the grouping. In their conclusion, they stated this methodology becomes increasing unscalable as the number of users and items increase . Johnson et al. [2004] took this further by analyzing the pattern across multiple sessions of the same user. They confirmed that users who visit the sites more frequently are more likely to make a purchase. Also clustering and classification of clickstreams can be found in Aggarwal et al. [2003] and Wei et al. [2012]. Park and Fader [2004] analyses web data from a marketing perspective in order to predict purchase for users. Cadez et al. [2003] was the first who clustered clickstream data by using mixture of first order Markov models . Some recent work using the multivariate t-distribution for modelbased clustering has been put forth by McLachlan and Krishnan [2007], and Andrews and Mcnicholas [2011] . Moreover Scholz et al. [2016] create a software package for modeling lists of clickstreams as zero-, first- and higher-order Markov chains and making cluster classification.

In path navigation analysis, since the number of distinct web-pages can be very high, one problem was the increase in the number of model's parameters. In order to reduce the number of parameters in the model Melnykov [2016] developed a methodology for grouping states according their similarity. A more precise approach extending the research of Melnykov [2016] for unsupervised and semi-supervised learning of clickstream data was from Gallaugher and McNicholas [2018].

They introduced a mixed of first order continuous Markov chain models for analyzing clickstream data. Their results compared with the discrete time model showed that the continuous time model is more accurate. They evaluate their results with simulations. We use the same methodology in the Gallaugher and McNicholas [2018] and according Albert et al. [1962], who considered the estimation of the infinitesimal generator in a continuous time Markov model.

1.1.3 Purpose of this study

Particularly we are interested in discovering common navigation patterns for the different users and discovering their group structure based in their internet behavior. So we treat clickstreams as sequences (of sites or web pages) and we are trying to cluster them. Because of the nature of the data we use Markov chains. Clickstream data are time-dependent and these data are traditionally modeled by means of Markov Chains(Cadez et al. [2003]). Also we face high dimensional data , these automatically leads in high number of model parameters. Clustering performance can be affected when we deal with high number of parameters. For this reason the proposed method includes straightforward selection algorithms such as EM algorithm (Dempster et al. [1977]) and inference criterion such as BIC (Schwarz et al.

[1978]).

Chapter 2

Markov Chains and Transition Functions

2.1 Continuous-Time Markov Chains

2.1.1 Definition and the minimal construction of a Markov chain

Many processes one may wish to model occur in continuous time (e.g. disease transmission events, cell phone calls, mechanical component failure times). A discrete-time approximation may or may not be adequate. A stochastic process in continuous time is a family, $X(t)_{t\geq 0}$, of random variables indexed by the positive real line $[0,\infty)$. The possible values of $X(t)_{t\geq 0}$, are referred to as the state space, S, of the process.

Definition (Homogeneous Markov chain in continuous time)

A continuous-time Markov chain on a finite or countable set, S, is a family of random variables, $X(t)_{t\geq 0}$, on a probability space $(\Omega, \mathcal{F}, P)$ such that

$$\begin{aligned}
&P(X(t_{n+1})=j|X(t_n)=i, X(t_{n-1})=i_{n-1}, ..., X(t_0)=i_0)\\
&=P(X(t_{n+1})=j|X(t_n)=i)\\
&=P_{i,j}(t_{n+1}-t_n)
\end{aligned}$$

for $j, i, i_{n-1}, ..., i_0 \in S$ and $t_{n+1} > t_n > ... > t_0 \geq 0$. The distribution of the Markov chain is determined by

$$\begin{aligned} \phi(i) &= P(X(0) = i) \leftarrow \text{initial distribution} \\ P_{i,j}(t) &= P(X(t+s) = j | X(s) = i) \leftarrow \text{transition probabilities} \end{aligned}$$

through the identity

$$\begin{aligned} &P(X(t_{n+1}) = j, X(t_n) = i, X(t_{n-1}) = i_{n-1}, ..., X(t_0) = i_0) \\ &= P_{i,j}(t_{n+1} - t_n) P_{i_{n-1},i}(t_n - t_{n-1}) \cdot ... \cdot P_{i_0,i_1}(t_1 - t_0)\phi(i_0) \end{aligned}$$

The transition probabilities $P(X(t_{n+1}) = j | X(t_n) = i)$ are assumed to depend only on the time difference $t_{n+1} - t_n$. The distribution of a Markov chain in continuous time is completely determined by the initial distribution and the transition probabilities. This construction establishes a unique parametrization of a CTMC by transition intensities. The transition probability for a continuous time Markov Model depends on time and satisfy the Chapman-Kolmogorov equations.

2.1.2 Transition probabilities for finite state space

For a continuous-time Markov chain on a finite state space the backward differential equation may be expressed in matrix form as:

$$P'(t) = QP(t) = P(t)Q$$

, with $P(0) = I$,

where $P(t) = (P_{i,j}(t))_{i,j} \in S$. Using the boundary condition $P(0) = I$ it turns out that the transition probabilities may expressed in terms of exponential matrices as

$$P(t) = \exp tQ = \sum_{n=0}^{\infty} \frac{(tQ)^n}{n!}$$

.

Using the forward and backward differential equations it may be possible to find closed form expressions for some of the transition probabilities,$P_{i,j}(t)$, for certain values of i, j.

Q is a square matrix whose value is constant in time and its called the infinitesimal generator of the process. Also $Q = q_{ij}$ where q_{ij} are the transition intensities.

2.1.3 The infinitesimal generator matrix

For a continuous-time Markov chain, $(X(t))_{t\geq 0}$, the transition intensities may be obtained from transition probabilities $P(t) = (P_{i,j}(t))_{i,j\in S}$ as the limits

$$\lim_{t\to 0^+} \frac{P_{i,i}(t) - 1}{t} = q_{i,i}$$
$$\lim_{t\to 0^+} \frac{P_{i,j}(t)}{t} = q_{i,j},\ i \neq j$$

Based on these assumptions derives the following theorem.

Theorem 1 *(a) Let*

$$q(i,j) = \begin{cases} -q(i) = -\sum_{j\neq i} q(i,j) & \textit{if } i = j, \\ q(i,j) & \textit{ifi} \neq j, \end{cases}$$

and let Q be the $(M \times M)$ matrix whose (i,j)th entry is q(i,j). The matrix of transition probability functions is given by $P(T) = \exp tQ$.

(b) $P[Z(t) = i, \qquad t_0 \leq t \leq t_0 + \alpha | Z(t_0) = i] = \exp -q(i)\alpha$ for all non-negative t_0 and α.

(c) If $Z(t_0) = i$ and $q(i) > 0$, there is,with probability one , a sample function discontinuity for some $t > t_0$, and in fact a first discontinuity which is a jump. If $0 < a \leq \infty$, the conditional probability that the first discontinuity in $[t_0, t_0 + a)$ is a jump to j , given that $Z(t_0) = i$ and there is a discontinuity in $[t_0, t_0 + a)$, is a $q(i,j)/q(i)$.

(d) Almost all sample functions are step functions with a finite number of jumps in any finite time interval.

2.1.4 The space of sample functions

According the Theorem 1 a sample function can be specified if we know the number of jumps made in $[0, T]$, the ordered lengths of time between jumps , and the succession of values taken on by the process in $[0, T]$.

We define the following random variables:

$$\tau_0(\omega) = 0, \qquad \tau_i(\omega) = \begin{cases} \text{The time at which the } i\text{th jump occurs if } \omega \in \Omega \\ +\infty \text{ otherwise} \end{cases}$$

$$T_i(\omega) = \begin{cases} \tau_{i+1}(\omega) - \tau_i(\omega) & \text{if } \tau_i(\omega) < \infty \\ 0 & \text{otherwise} \end{cases}$$

$$N(T, \omega) = \text{ The largest integer } n \text{ , for which } \tau_n(\omega) < T,$$

$$Z_i(\omega) = Z(\tau_i(\omega), \omega) \qquad i = 0, 1, 2, \cdots .$$

Where $T_i(\cdot)$ is the time spent in the ith state ,$N(T, \cdot)$ is the number of jumps made by the process on $[0, T]$, and $Z_i(\cdot)$ is the state of the process immediately after the ith jump. In fact with probability one , a sample function of $\{Z(t, \cdot), 0 \leq t < T\}$ can be represented as an ordered sequence: $\{Z(t, \omega), 0 \leq t < T\}$=

$$((Z_0(\omega), T_0(\omega)), \cdots , (Z_{N(T,\omega)-1}(\omega), T_{N(T,\omega)-1}(\omega)), Z_{N(T,\omega)}).$$

This means that:

$$Z(t, \omega), 0 \leq t < T = ((z_0, t_0), \cdots , (z_{n-1} t_{n-1}), z_n)$$

, and thus the path function starts at z_0 at time zero ,remains in z_0 for t_0 units of time , makes a jump to z_1 ,remains in z_1 for t_1 units of time, $\cdots$, jumps to z_{n-1},remains there for t_{n-1} units of time and then makes the final jump to z_n, and remains there at least until time T. The probability distribution on the space of sample functions is described from the following theorem:

Theorem 2 *Let*

$$q'(i, j) = \begin{cases} 0 & if\ i = j \\ q(i, j) & if\ i \neq j \end{cases}$$

Then $Pr[N(T) = n \& Z_0 = z_0 \& T_0 \leq \alpha_0 \& \cdots \& Z_{n-1} = z_{n-1} \& T_{n-1}$

$$\leq \alpha_{n-1} \& Z_n = z_n] = Pr[Z(0) = z_0] \exp -q(z_n)T \cdot \int_{S_n} \prod_{j=0}^{n-1} dt_i q'(z_i, z_{i+1}) \exp -[q(z_i) - q(z_n)] t_j$$

where

$$S_n = \left\{ (t_0, t_1, \cdots, t_{n-1}) : \sum_{j=0}^{n-1} t_j < T \& 0 \leq t_j \leq \alpha_j \right\} \text{ if } n > 0.$$

$$Pr[N(T) = 0 \& Z_0 = z_0] = Pr[Z(0) = z_0] \exp -q(z_0)T.$$

Sufficient statistics and the maximum likelihood

If k independent realizations $v_1, v_2, \cdots, v_k$ of $\{Z(t), 0 \leq t < T\}$ are observed. The likelihood function is defined by the following equation:

$$\mathcal{L}_{\mathcal{Q}}^{(k)} = \prod_{j=1}^{k} f_Q(v_j).$$

If we let $N_T^{(K)}(i,j)$ = the total number of transitions from state i to state j observed during the k trials ans $A_T^{(K)}(i)$ = the total length of time that state i is occupied during the k trials , the log-likelihood is written :

$$\log \mathcal{L}_{\mathcal{Q}}^{(k)} = C_k + \sum_i \sum_{j \neq i} N_T^{(k)}(i,j) \log q(i,j) - \sum_i A_r^{(k)}(i) q(i),$$

C_k is finite with probability one and its independent from Q. The set $\{N_T^{(k)}(i,j), A_T^{(k)}(i)\}_{j \neq i}$ is a sufficient statistic for Q. The maximum likelihood estimates (m.l.e.) for $q(i,j)$ are seen to be :

$$\hat{q}_T^{(k)}(i,j) = \frac{N_T^{(k)}(i,j)}{A_T^{(k)}(i)} \qquad \text{if } i \neq j \ \& \ A_T^{(k)}(i) > 0.$$

If $A_T^{(k)} = 0$, the m.l.e does not exist so is made the convention that

$$\hat{q}_T^{(k)}(i,j) = 0 \text{ if } i \neq j \text{ and } A_T^{(k)}(i) = 0.$$

Example

Let's assume three possible states $S = 1, 2, 3$. There are the following data paths $D = d_1, d_2, d_3, d_4$. In parenthesis the time spent at the state before transiting to the next.

$$\begin{aligned}
d_1 &= 1(3min) \rightarrow 2(3min) \rightarrow 3\\
d_2 &= 1(8min) \rightarrow 2(12min) \rightarrow 3\\
d_3 &= 1(2min) \rightarrow 3\\
d_4 &= 2(2min) \rightarrow 3
\end{aligned}$$

Problem: What is the probability of transiting from state i to state j given that s time has elapsed since entering state i?

Solution:

First fit a continuous time Markov chain model to the data by estimating the (infinitesimal) generator Q.

The general form for the generator of a continuous time Markov chain with three states is

$$Q = \begin{bmatrix} -\alpha_1 & \alpha_{12} & \alpha_{13} \\ \alpha_{21} & -\alpha_2 & \alpha_{23} \\ \alpha_{31} & \alpha_{32} & -\alpha_3 \end{bmatrix} \tag{2.1}$$

Where the diagonal elements satisfy $\alpha_i = \sum_{j \neq i} \alpha_{ij}$. The easiest way for estimating Q may be using the maximum likelihood estimate (m.l.e): Metzner [2007] we know that the m.l.e for $\alpha(ij)$ is given by

$$\hat{\alpha}_{ij} = \frac{N_{ij}}{T_i}$$

where N_{ij} is the number of observed transitions from i to j and T_i is the total time spent in state i by the observed process.

So the estimates are $\hat{\alpha}_{12} = 2/(3+8+2) = 0.154, \hat{\alpha}_{13} = 1/(3+8+2) = 0.077, \hat{\alpha}_{21} = 0/(2+3+12) = 0, \hat{\alpha}_{23} = 3/(2+3+12) = 0.176$.

The paths lists no observations for the system being in state 3, so no estimates of $\hat{\alpha}_{31}$ and $\hat{\alpha}_{32}$ are possible. If the system has been observed to stay in state 3 for a time without making a transition, then T_3 is positive and we would get the m.l.e s $\hat{\alpha}_{31} = \hat{\alpha}_{32} = 0$. In this case, state 3 would be absorbing (represented by a row of zeros in the generator) and the generator would be the one given above.

Finally, using the Kolmogorov backward equations, it is possible to compute the transition probabilities for the fitted model. We know that the transition matrix over a time interval of length s is given by

$$P(s) = \exp(sQ)$$

2.2 Mixture Models

2.2.1 From Factor Analysis to Mixture Models

Mixture model or,in literature,mixture distributions are convex combinations of ”component” distributions. In statistics are typically used to generalize distributional assumptions and where a heterogeneous population consists of several sub populations. In the simplest case mixture models are considered as flexible tools for achieving a good fit to data. We present some important topics for mixture models based in Smyth [1997] and Böhning and Seidel [2003]

More formally , we say that if we have a data set

$$D = \{\underline{x}_1, ..., \underline{x}_N\}$$

where $\underline{x}_i$ is a d-dimensional vector measurement. The points are generated from an underlying density $p(\underline{x})$.

A flexible model for $\underline{x}_i$ is a finite mixture model with K components:

$$p(\underline{x}_i|\Theta) = \sum_{k=1}^{K} p_k(\underline{x}_i|z_{i_k} = 1, \theta_k)a_k$$

where:

- The $p_k(\underline{x}_i|z_{i_k} = 1, \theta_k)$ are mixture components , $1 \leq k \leq K$. Each is a density or distribution defined over $p(\underline{x}_i)$, with parameters θ_k. Each of the components can be any any distribution pr density function , and moreover its not necessary for the components to have all the same form.

- $z_i = (z_{i_1}, ..., z_{i_K})$ is a vector of K binary indicator variables , where z_i plays the role of an indicator random variable which represents the identity of the mixture component that generated x_i.
- The $a_k = p(z_{i_k} = 1)$ are the mixture weights , representing the probability that a randomly selected $\underline{x_i}$ was generated by component k,where $\sum_k a_k = 1$.

The over-all parameter vector of the mixture model with K components is thus

$$\Theta = \{a_1, ..., a_K, \theta_1, ..., \theta_K\}$$

.

This is a complete stochastic model, so it gives us a recipe for generating new data points: first pick a distribution, with probabilities given by the mixing weights, and then generate one observation according to that distribution.

Symbolically,

$$Z \sim \text{mult}(a_1, a_2, ..., a_k)$$

$$X|Z \sim f_z$$

where the discrete random variable Z says which component X is drawn from. As it concerns what kind of distribution the $f_k s$ are,in principle, we could make these completely arbitrary, and we'd still have a perfectly good mixture model. In practice, a lot of effort is given over to parametric mixture models, where the f_k are all from the same parametric family, but with different parameters. For instance they might all be Gaussians with different centers and variances, or all Poisson distributions with different means, or all power laws with different exponents. (It's not strictly necessary that they all be of the same kind.) The parameter vector, of the kth component is θ_k , so the model becomes

$$f(x) = \sum_{k=1}^{K} a_k f(x; \theta_k)$$

$$\theta = (a_1, a_2, ...a_k, \theta_1, \theta_2, ...\theta_k)$$

.

When $K = 1$, we have a simple parametric distribution, and density estimation reduces to estimating the parameters,by maximum likelihood . On the other hand when $K = n$, the number of observations, we have gone back towards kernel density estimation.

Many components in the mixture lets us fit many distributions very accurately, with low approximation error or bias. The cost is that we have more parameters and so we can't fit any one of them as precisely, and there's more variance in our estimates.

Geometry

If we use a mixture model with q +1 components, we will also get data which clusters around a q-dimensional plane. Furthermore, by adjusting the mean of each component, and their relative weights, we can make the global mean of the mixture whatever we like. And we can even match the covariance matrix of any q-factor model by using a mixture with q +1 components . Although this mixture distribution will hardly ever be exactly the same as the factor model's distribution(e.g. mixtures of Gaussian aren't Gaussian), the mixture will usually (but not always) be multimodal while the factor distribution is always unimodal — but it will have the same geometry, the same mean and the same covariances.

2.2.2 Identifiability

Two kinds of identification problems are common for mixture models; one is trivial and the other is fundamental. The trivial one is that we can always swap the labels of any two components with no effect on anything observable at all — if we decide that component number 1 is now component number 7 and vice versa, that doesn't change the distribution of X at all. This label degeneracy can be annoying, especially for some estimation algorithms, but that's the worst of it. A more fundamental lack of identifiability happens when mixing two distributions from a parametric family just gives us a third distribution from the same family. For example, suppose we have a single binary feature, say an indicator for whether someone will pay back a credit card. We might think there are two kinds of customers, with high- and low-risk of not paying, and try to represent this as a mixture of Bernoulli distribution. If we try this, we'll see that we've gotten a single Bernoulli distribution with an intermediate risk of repayment. A mixture of Bernoulli is always just another Bernoulli. More generally, a mixture of discrete distributions over any finite number of categories is just another distribution over those categories.

2.2.3 Probabilistic Clustering

Another way of view mixture models ,is a way of putting similar data points together into “clusters”, where clusters are represented by, precisely, the component distributions. The idea is that all data points of the same type, belonging to the same cluster, are more or less equivalent and all come from the same distribution, and any differences between them are matters of chance. One of the very nice things about probabilistic clustering is that actually claims something about what the data looks like; it says that it follows a certain distribution. We can check whether it does, and we can check whether new data follows this distribution. If it does, great; if not, if the predictions systematically fail, then the model is wrong. We can compare different probabilistic clusters by how well they predict. The best number of clusters to use is the number which will best generalize to future data. If we don’t want to wait around to get new data, we can approximate generalization performance by cross-validation, or by any other adaptive model selection procedure.

2.2.4 Classification-semi supervised learning

In machine learning, a common scenario is to have a small amount of labeled and a large amount of unlabeled data. For example, it may be that we have access to many images of faces; however, only a small number of them may have been labeled as instances of known faces. In semi-supervised learning, one tries to use the unlabeled data to make a better classifier than that based on the labeled data alone. This is a common issue in many examples since often gathering unlabeled data is cheap (taking photographs, for example). However, typically the labels are assigned by humans, which is expensive

2.2.5 Learning Mixture Models from Data

To fit a mixture model to data we can use maximum likelihood. Assuming that the data points $\underline{x}_i$ are conditionally independent given the model and its parameters Θ , we have (as always, we'll use the logarithm to turn multiplication into addition)

$$L(\Theta) = \sum_{i=1}^{N} \log(\sum_{k=1}^{K} p_k(\underline{x}_i | z_{i_k} = 1, \theta_k) a_k$$

where $a_k = p(z_{i_k} = 1)$ is the marginal probability that a randomly selected $\underline{x}_i$ was generated by component k. If we take partial derivatives of this log-likelihood and set them to 0 we get a set of coupled non=linear equations. For example if we suppose that the component parameters θ_k were known and we were just learning the $a_k's$,we have

$$\frac{\partial l(\Theta)}{\partial \alpha_j} = \sum_{i=1}^{N} \frac{p_j(\underline{x_i} | z_{ij} = 1, \theta_j)}{\sum}$$

Setting these to 0 we get K non-linear equations. A valid approach for learning the parameters of mixture models is by using Gradient Desent. In order to decrease the cost of having to set learning rates another widely-used alternative is the Expectation-Maximization(EM) algorithm. We describe the EM algorithm detailed in next chapter and we adjust it for our model.

2.2.6 A Mixture of First Order Continuous Time Markov Models

Definition of the model

Our approach is based on a mixture of first order Continuous Time Markov Chain Model. Consider a website consisting of different web pages. These web pages include J different categories. The click-stream of interest is defined by the transitions from one category to another(or from a state to another). Suppose N click-streams are observed from a population with G types and we take into account the amount of time spent in each category. Denote them by

$$X_i = (x_{i_1}, x_{i_2}, ..., x_{iL_i})$$

and the observed times which the chain remains in each category

$$T_i = (t_{i_1}, t_{i_2}, ..., t_{iL_i})$$

correspondingly.

The one step transition matrix for group g is

$$\mathbf{\Lambda}_g = \begin{pmatrix} \lambda_{g11} & \lambda_{g12} & \cdots & \lambda_{g1J} \\ \lambda_{g21} & \lambda_{g22} & \cdots & \lambda_{g2J} \\ \vdots & \vdots & \ddots & \vdots \\ \lambda_{gJ1} & \lambda_{gJ2} & \cdots & \lambda_{gJJ} \end{pmatrix}.$$

Now, define the initial probabilities

$$\alpha_{gx_{i1}} = P(X_{i1} = x_{i1} | \mathbf{X}_i \text{ is in group } g),$$

for $i = 1, 2, \ldots, N$. For ease of notation, and recalling that $x_{i1} \in \{1, 2, \ldots, J\}$, denote an initial probability vector for each group g by $\boldsymbol{\alpha}_g = (\alpha_{g1}, \alpha_{g2}, \ldots, \alpha_{gJ})$. Finally, for ease of notation, we denote the total number of transitions from state j to state k for clickstream i by n_{ijk} .

the generator matrix is the following:

$$\mathbf{Q}_g = \begin{pmatrix} q_{g11} & q_{g12} & \cdots & q_{g1J} \\ q_{g21} & q_{g22} & \cdots & q_{g2J} \\ \vdots & \vdots & \ddots & \vdots \\ q_{gJ1} & q_{gJ2} & \cdots & q_{gJJ} \end{pmatrix},$$

where $q_{gjk} \geq 0$ for $j \neq k$ and $q_{gjj} = -\sum_{k \neq j} q_{gjk}$ for $g \in \{1, 2, \ldots, G\}$. The first item we note here is that the underlying transition probabilities are given by

$$P(X_{i(l+1)} = x_{i(l+1)} | X_{il} = x_{il}, z_{ig} = 1) = -\frac{q_{gx_{il}x_{i(l+1)}}}{q_{gx_{il}x_{il}}}.$$

The second is that each T_{il} are independent and

$$T_{il} | (X_{il} = x_{il}, z_{ig} = 1) \sim \text{Exp}(-q_{gx_{il}x_{il}}),$$

where $\mathrm{Exp}(a)$ denotes an exponential distribution with rate a. We now derive the joint density of an observed clickstream $\mathbf{x}$ with length L and time vector $\mathbf{t}$ for a single component.

$$\begin{aligned}
f(\mathbf{x},\mathbf{t}|\boldsymbol{\alpha},\mathbf{Q}) &= f(x_1,t_1,x_2,t_2,\ldots,x_L,t_L) \\
&= f(x_1)f(t_1|x_1)f(x_2|t_1,x_1)f(t_2|x_2,t_1,x_1)\times\cdots\times f(x_L|\mathcal{F})f(t_L|\mathcal{F}\cup\{x_L\}) \\
&= f(x_1)f(t_1|x_1)f(x_2|x_1)f(t_2|x_2)\times\cdots\times f(x_L|x_{L-1})f(t_L|x_L) \\
&= \left(\prod_{j=1}^{J}\alpha_j^{I(x_1=j)}\right)\left[\prod_{l=1}^{L-1}(-q_{x_lx_l})\exp\{q_{x_lx_l}t_l\}\left(\frac{q_{x_1,x_{l+1}}}{-q_{x_lx_l}}\right)\right]\times(-q_{x_Lx_L})\exp\{q_{x_Lx_L}t_L\} \\
&= \left(\prod_{j=1}^{J}\alpha_j^{I(x_1=j)}\right)\exp\left\{\sum_{j=1}^{J}\sum_{l=1}^{L}q_{jj}t_lI(x_l=j)\right\}\left[\prod_{j=1}^{J}\prod_{k\neq j}^{J}q_{jk}^{n_{jk}}\right]\left[-\prod_{j=1}^{J}q_{jj}^{I(x_L=j)}\right]
\end{aligned}$$

We denote the initial probability vector by $\boldsymbol{\alpha}_g$, as before. The likelihood is the following:

$$\mathcal{L}_{\text{obs}}(\boldsymbol{\vartheta}|\mathcal{D}_{\text{o}}) =$$
$$\prod_{i=1}^{N}\prod_{g=1}^{G}\left[\pi_g\left(\prod_{j=1}^{J}\alpha_{gj}^{I(x_{i1}=j)}\right)\left(\prod_{j=1}^{J}\prod_{k\neq j}^{J}q_{gjk}^{n_{ijk}}\right)\left(-\prod_{j=1}^{J}q_{gjj}^{I(x_{iL_i}=j)}\right)\exp\left\{\sum_{j=1}^{J}\sum_{l=1}^{L}q_{gjj}t_{il}I(x_{il}=j)\right\}\right]^{z_{ig}}$$

and the complete-data log-likelihood is

$$\begin{aligned}
\ell_{\text{c}} = \sum_{i=1}^{N}\sum_{g=1}^{G}\Bigg[z_{ig}\Bigg(&\log\pi_g+\sum_{j=1}^{J}I(x_{i1}=j)\log\alpha_{gj}+\sum_{j=1}^{J}\sum_{k\neq j}^{J}n_{ijk}\log q_{gjk} \\
&+\sum_{j=1}^{J}I(x_{iL_i}=j)\log(-q_{gjj})+\sum_{j=1}^{J}\sum_{l=1}^{L_i}q_{gjj}t_{il}I(x_{il})\Bigg)\Bigg].
\end{aligned}$$

The next step is to apply the EM algorithm. We used the EM algorithm as it is described in Biernacki et al. [2003].

Chapter 3

EM algorithm

The EM algorithm is an iterative algorithm for doing ”local ascent” of the likelihood (or log-likelihood) function. Its is usually east to implement , it enforces parameter constraints automatically , and it does not require the specification of a step-size(the step size is implicit at each interaction of the algorithm). Moreover EM is a general procedure and can be applied in any problems where there is missing data. In case of mixture models the missing data are the z indicators for component membership.

3.1 Outline the EM algorithm for mixture models

EM algorithm is a generalization of k-means algorithm. In EM there are two steps called expectation step, which is equivalent to assigning points to the clusters and maximization step, which is equal to recomputing centroids.
EM algorithm starts from an initial estimate of the parameter set Θ (e.g. random initialization) , and then proceeds to iteratively update Θ until convergence is detected. The E- step and M-step are consisted in each iteration of the algorithm.

In the E-step the algorithm computes the expected log=likelihood with respect to the probability of the $z_i's$ conditioned on the $\underline{x}_i's$ and the current values of the parameters. For mixture models the expected value $E[z_{i_k}] = p(z_{i_k} = 1|\underline{x}_i, \Theta)$. We compute all N data points for each of the K clusters , using Bayes rule.

In the M-step the algorithm computes new parameter values that maximize the expected log-likelihood ,given the NxN matrix of $p(z_{i_k} = 1|\underline{x}_i, \theta)$ values produced by the E-Step.

Both E-Step and M-Step are computed straightforward for mixture models , and that's why the EM algorithm is very popular in practice for fitting mixture models.

3.2 The E-Step for Mixture Models

In the E-Step,given a set of parameters Θ ,we compute the "weight" of data points $\underline{x}_i$ in component k as

$$w_{i_k} = p(z_{i_k} = 1|\underline{x}_i, \Theta) = \frac{p_k(\underline{x}_i|z_k, \theta_k)a_k}{\sum_{m=1}^{K} p_m(\underline{x}_i|z_m, \theta_m)a_k}$$

This follows from the direct application of Bayes rule. For mixture models with different components , the components would have different functional forms , but the general equation for computing mixture weights has the same general form. These weights are an $N \times K$ matrix where each row sums to 1 and contains the weights for data vector $\underline{x}_i$.
We assume that each $\underline{x}_i$ was generated by a single component ,so these probabilities reflect the uncertainty about which component came from.

3.3 The M-Step for Mixture Models

Given the weights from the E-Step we use the weights and the data to calculate new parameter values. Suppose $N_k = \sum_{i=1}^{N} w_{i_k}$ the sum of the weights for the kth component this is the effective number of data points assigned to component K.

The new estimate of the mixture weights is $a_k^{new} = \frac{N_k}{N}, 1 \leq k \leq K$. These are the new mixture weights, with $\sum_{k=1}^{K=1} \alpha_k = 1$. After that the updated mean is calculated is a similar way of manner of how a standard empirical average is calculated. The difference is that, ith measurement $x(i)$ has a fractional weight ω_{ik}.

After all of the new parameters are updated , the M-step is complete and we can now go back and recompute the weights in the E-step, then recompute the parameters again in the E-step, and continue updating in this order. Each pair of E and M steps is considered to be one iteration.

3.4 EM algorithm for the Mixture Markov Model

In the E step, we update the indicator variables z_{ig}. At iteration $s+1$, this update is given by

$$\hat{z}_{ig} = \frac{h(\hat{\pi}_g, \hat{\boldsymbol{\alpha}}_g, \hat{\mathbf{Q}}_g, \mathbf{x}_i, \mathbf{t}_i)}{\sum_{g'=1}^{G} h(\hat{\pi}_{g'}, \hat{\boldsymbol{\alpha}}_{g'}, \hat{\mathbf{Q}}_{g'}, \mathbf{x}_i, \mathbf{t}_i)}, \tag{3.1}$$

where

$$\begin{aligned} h(\hat{\pi}_g, \hat{\boldsymbol{\alpha}}_g, \hat{\mathbf{Q}}_g, \mathbf{x}_i, \mathbf{t}_i) = \hat{\pi}_g \left(\prod_{j=1}^{J} (\hat{\alpha}_{gj})^{I(x_{i1}=j)} \right) \left[\prod_{j=1}^{J} (-\hat{q}_{gjj})^{I(x_{iL}=j)} \right] \left[\prod_{j=1}^{J} \prod_{k \neq j}^{J} (\hat{q}_{gjk})^{n_{ijk}} \right] \\ \times \exp\left\{ \sum_{j=1}^{J} \sum_{l=1}^{L_i} \hat{q}_{gjj} t_{il} I(x_{il}=j) \right\} \end{aligned}$$

At the M step, we update our parameters, $\hat{\pi}_g$, $\hat{\boldsymbol{\alpha}}_g$ and $\hat{\mathbf{Q}}_g$. The updates for $\hat{\mathbf{Q}}_g$, are given by

$$\hat{q}_{gjk} = \begin{cases} \frac{\sum_{i=1}^{N} z_{ig} n_{ijk}}{\hat{\lambda}_{gj}} & \text{if } j \neq k, \\ -\sum_{k \neq j} \hat{q}_{gjk} & \text{if } k = j, \end{cases}$$

where $\hat{\lambda}_{gj} = a/b$ with

$$\begin{aligned} a &= \sum_{i=1}^{N} \left(\sum_{l=1}^{L_i} \hat{z}_{ig} t_{il} I(x_{il}=j) + \sum_{k \neq j}^{J} \hat{z}_{ig} n_{ijk} \right), \\ b &= \sum_{i=1}^{N} \hat{z}_{ig} I(x_{iL_i}=j) + \sum_{i=1}^{N} \sum_{k \neq j}^{J} \hat{z}_{ig} n_{ijk}. \end{aligned}$$

At the M step, we update our parameters, $\hat{\pi}_g$, $\hat{\boldsymbol{\alpha}}_g$ and $\hat{\mathbf{Q}}_g$. We also update each $\hat{\mathbf{Q}}_g$.

3.5 Initialization and Convergence Issues for EM

The EM algorithm can be started by either weight initialization or with parameter initialization. The initial parameters or weights can be chosen randomly or could be chosen via some heuristic method (such as by using the k-means algorithm to cluster the data first and then defining weights based on k-means memberships). Convergence is generally detected by computing the value of the log-likelihood after each iteration and halting when it appears not to be changing in a significant manner from one iteration to the next.

For avoiding computational issues , as discussed in Gallaugher and McNicholas [2018] and Melnykov [2016], like the cases where there are no transitions present in the data between two states, i.e.,$n_{ijk} = 0$, for all we penalized likelihood. For the estimates q_{gjk} which would be zero for all g we set a lower bound of 10^{-6} for all parameters. An estimation of zero value would be a problem, because causes problems with the calculation of the likelihood. So we assume that all the states communicate with each other.

3.6 Selecting the number of clusters

Selecting the number of clusters is quite a sensitive topic. For finite mixtures, a suitable number of components can be selected using different criteria. Information criteria such as the Akaike information criterion (AIC) or Bayesian information criterion (BIC)Schwarz et al. [1978] have been used in a model-based clustering context where it has been shown for the BIC that the number of components are consistently selected under certain conditions, in particular ensuring that the component densities remain bounded Keribin [2000]. We used BIC for selecting the number of groups for our model where is defined:

$$\mathrm{BIC} = 2\ell_{\mathrm{obs}}(\boldsymbol{\vartheta}|D) - p \log N$$

Chapter 4

Data Description

We first make a full description of our data, how it is collected, the data processing which is needed for conducting an exploratory data analysis to fit the model we have already introduce. Our data is derived from a dataset that was constructed by YOOCHOOSE GmbH to support participants in the RecSys Challenge 2015. The YOOCHOOSE dataset contains a collection of sessions from a retailer, where each session is encapsulating the click events that the user performed in the session. For some of the sessions, there are also buy events; means that the session ended with the user bought something from the web shop. The data was collected during several months in the year of 2014, reflecting the clicks and purchases performed by the users of an on-line retailer in Europe. To protect end users privacy, as well as the retailer, all numbers have been modified.

This web shop monitored users usage. It records the users (sessions) id, the different categories they have visited, the exact date and time that they have made the visit and so derived from the previous, the amount of time that each user has spent in each category. The categories normally have discrete items, and there is also an special offer category that can be common for all items. However the category of the same item depends on the context where the click occurred, as described next in the fileds/format. For this reason it is preferred to use the category values instead of items, in our analysis.

A *session* is defined as a period of sustained web browsing or a sequence of page viewings. If a user has not viewed any pages for a considerable amount of time, it is assumed that the viewing session has ended and that the next page viewing marks the beginning of a new session. Thus, different sessions id may refer to the same user, but there is no definitive way of knowing that, and that's why for simplicity each session id is assumed to correspond to a unique user. In the second file we have the users id and if the user made a purchase or deferred. We combined those two files for our analysis.

The YOUCHOOSE dataset comprises two different files:

Click events. The file *yoochoose-clicks.dat* comprising the clicks of the users over the items.

Listing 4.1: Sample of *yoochoose-clicks.dat* records

```
Session_ID        Timestamp           Item_ID  Category
<dbl>            <dttm>              <dbl> <chr>
1      5003694  2014-06-23  19:11:22  214836868  S
2      5003694  2014-06-23  19:13:39  214836868  S
3      5003691  2014-06-23  18:30:47  214811756  4
4      5003691  2014-06-23  18:30:56  214811752  4
5      5003691  2014-06-23  18:30:57  214811754  4
6      5003691  2014-06-23  18:31:07  214839905  4
```

Each record/line in the file has the following fields and format organized per columns as comma separated values (csv):

1. Session ID – the id of the session. In one session there are one or many clicks. Could be represented as an integer number.

2. Timestamp – the time when the click occurred. Format of YYYY-MM-DDThh:mm:ss.SSSZ

3. Item ID – the unique identifier of the item that has been clicked. Could be represented as an integer number.

4. Category – the context of the click. The value "S" indicates a special offer, "0" indicates a missing value, a number between 1 to 12 indicates a real category identifier, any other number indicates a brand. E.g. if an item has been clicked in the context of a promotion or special offer then the value will be "S", if the context was a brand i.e BOSCH, then the value will be an 8-10 digits number. If the item has been clicked under regular category, i.e. sport, then the value will be a number between 1 to 12.

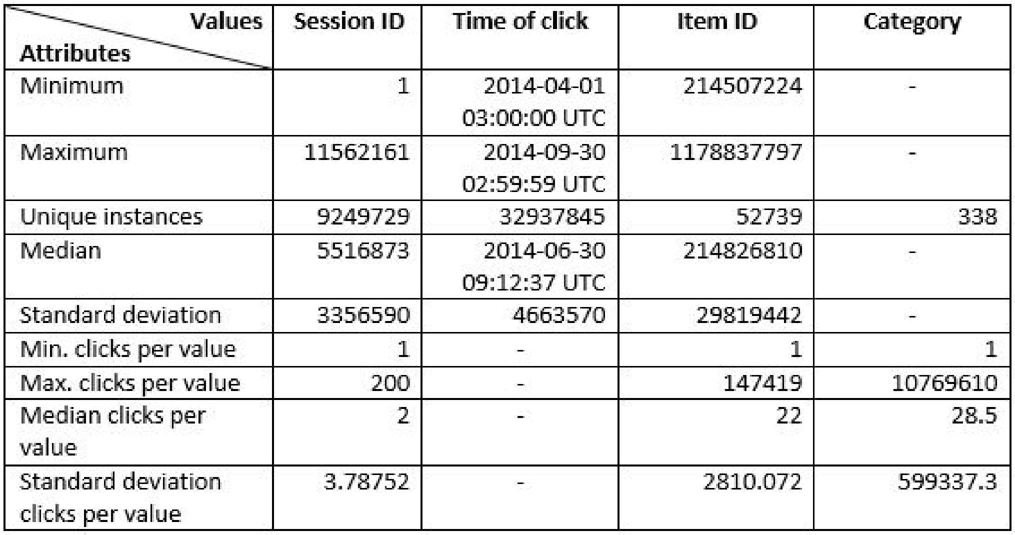
Buy events The file *yoochoose-buys.dat* comprising the buy events of the users over the items. Each record/line in the file has the following field and format organized per columns as comma separated values (csv)s:

1. Session ID - the id of the session. In one session there are one or many buying events. Could be represented as an integer number.
2. Timestamp - the time when the buy occurred. Format of YYYY-MM-DDThh:mm:ss.SSSZ
3. Item ID – the unique identifier of item that has been bought. Could be represented as an integer number.
4. Price – the price of the item. Could be represented as an integer number.
5. Quantity – the quantity in this buying. Could be represented as an integer number.

The Session ID in yoochoose-buys.dat will always exist in the yoochoose-clicks.dat file – the records with the same Session ID together form the sequence of click events of a certain user during the session, also referred as a path. The session could be short (few minutes) or very long (few hours), it could have one click or hundreds of clicks. All depends on the activity of the user.

4.1 Dataset analysis

To start the analysis, the dataset *yoochoose-clicks.dat* was read in **R**, i.e. in total 33003944 observations (clicks) described by four values each (*Session ID*, *Time of click*, *Item ID* and *Category*). The Category "0" is interpreted as a missing value(NA).
Clicks were collected in a period starting from 1st Aprin 2014 and finishing at 30th June 2014, from 9249729 unique sessions, created by users that browsed though 52739 different Items and 338 categories (Table 4.1). Sessions are generally short, as they consist of only two clicks in average. A simple way of visualization of our data is with histograms. Because of the large amount of data we changed the scale for presenting them in a simple way. In Figures 4.1, 4.2 and 4.3 are the histograms of frequency distributions for clicks per session,



Values / Attributes	Session ID	Time of click	Item ID	Category
Minimum	1	2014-04-01 03:00:00 UTC	214507224	-
Maximum	11562161	2014-09-30 02:59:59 UTC	1178837797	-
Unique instances	9249729	32937845	52739	338
Median	5516873	2014-06-30 09:12:37 UTC	214826810	-
Standard deviation	3356590	4663570	29819442	-
Min. clicks per value	1	-	1	1
Max. clicks per value	200	-	147419	10769610
Median clicks per value	2	-	22	28.5
Standard deviation clicks per value	3.78752	-	2810.072	599337.3

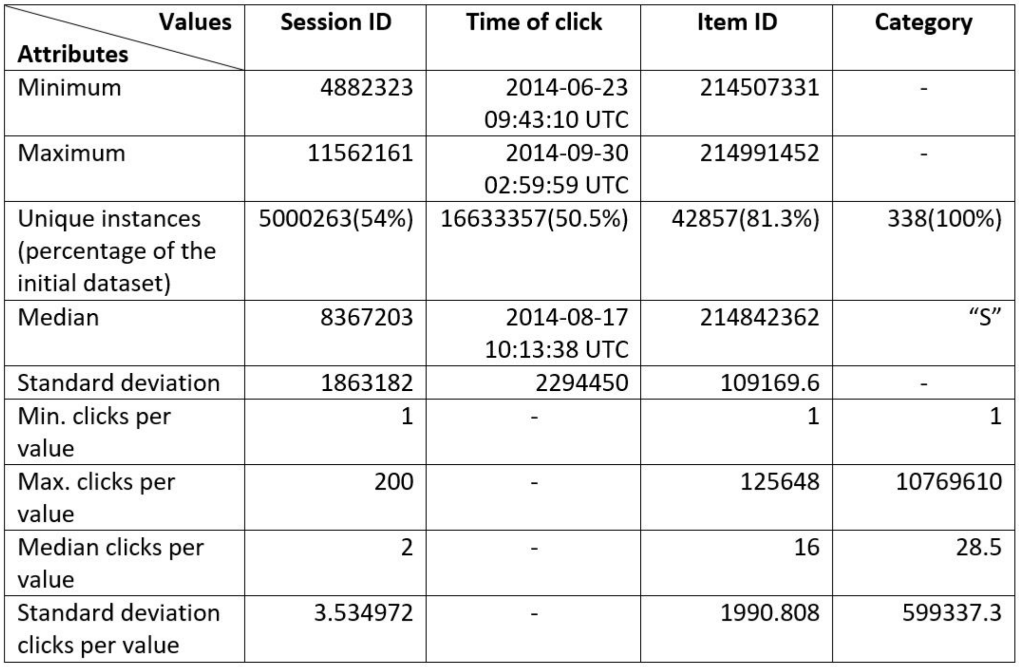
Table 4.1: Initial raw dataset *yoochoose-clicks.dat*.

per item and per category. For categories the vast majority of clicks are concentrated in a few categories. More specifically in figure 4.4 are presented the 13 most clicked categories, where it is clear that category ”S” attracts almost the majority of all clicks.

4.2 Data preparation

In our study, as an assumption, click events are the categories. So to clear the data, all the observations with missing values in the category column were removed and only the complete cases were retained. The proportion of the missing values is important, about 50% of the pages viewed by users were removed from the dataset. So the dataset remains with 16666291 observations (50.5% of initial clicks).
From table 4.2 and figures 4.5, 4.6 and 4.7, the following are observed: a) All observations between 1st April 2014 and 22nd June 2014 and about 45% of the sessions have missing values, b) no loss of information on the categories, c) the shape of frequency distributions is similar to the ones of original dataset.

Because of time and computational limitations, only 371984 of total 50000263 paths were calculated, as defined in ”Data Description”.



Values / Attributes	Session ID	Time of click	Item ID	Category
Minimum	4882323	2014-06-23 09:43:10 UTC	214507331	-
Maximum	11562161	2014-09-30 02:59:59 UTC	214991452	-
Unique instances (percentage of the initial dataset)	5000263(54%)	16633357(50.5%)	42857(81.3%)	338(100%)
Median	8367203	2014-08-17 10:13:38 UTC	214842362	“S”
Standard deviation	1863182	2294450	109169.6	-
Min. clicks per value	1	-	1	1
Max. clicks per value	200	-	125648	10769610
Median clicks per value	2	-	16	28.5
Standard deviation clicks per value	3.534972	-	1990.808	599337.3

Table 4.2: Dataset filtered - only complete cases, with no missing values.

Listing 4.2: Sample of paths

```
[[1]]"5003694" "S" "S"
[[2]]"5003691" "4" "4" "4" "4" "4" "4" "4"
[[3]]"5003684" "S" "S" "S"
[[4]]"5003681" "S" "S" "S" "S" "S"
[[5]]"5003649" "S" "S"
[[6]]"5003678" "S" "S"
```

In table 4.3, the time spent on any page and the time spent on any path is analyzed. It is impossible to derive anything meaningful from these sessions as the time the user spends viewing the page is not known. These times are normally calculated from the time differences between the request time of the current page and the request time of the next page viewed. Without the next page viewed, the time spent viewing the page cannot be calculated.

Listing 4.3: Time spent in each state of paths at Listing 4.2

```
[[1]]Time difference of 2.275167 mins
```

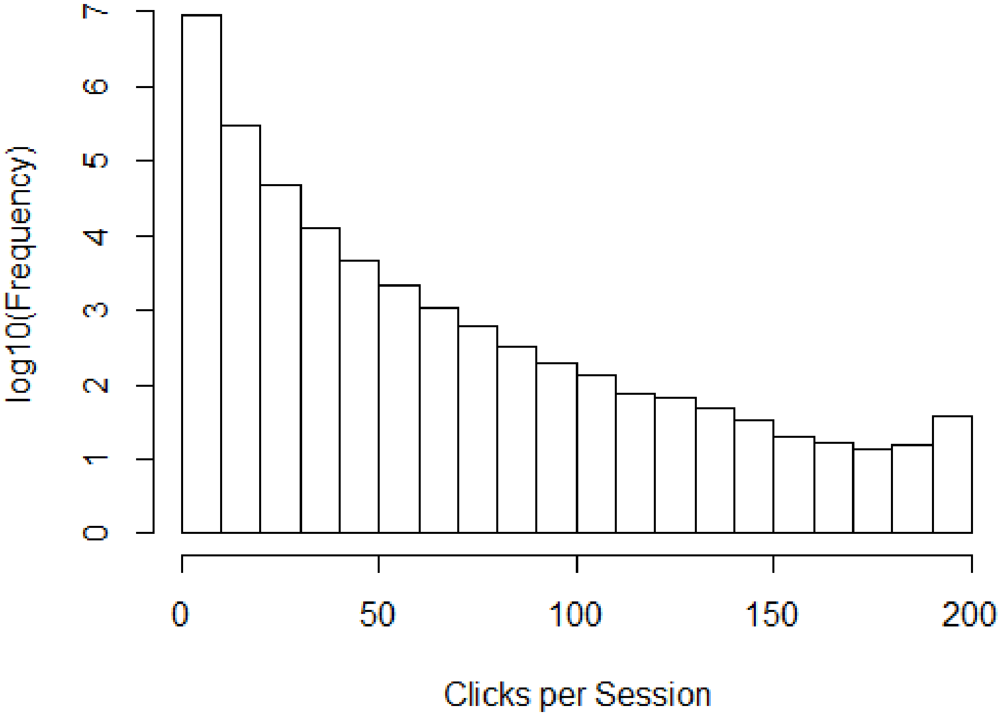


Figure 4.1: Frequency distribution of number of clicks per session in log scale at y-axis

```
[[2]]Time differences in secs
[1]      9.025      1.058     10.065  1332.401      1.278
938.985

[[3]]Time differences in secs
[1]  10.212  12.902

[[4]]  Time differences in secs
[1]  2657.288      7.537     29.092     37.713

[[5]]  Time difference of 1.72435 mins

[[6]]  Time difference of 55.923 secs
```

In the dataset we entered a minimum bound of time spent in each state,

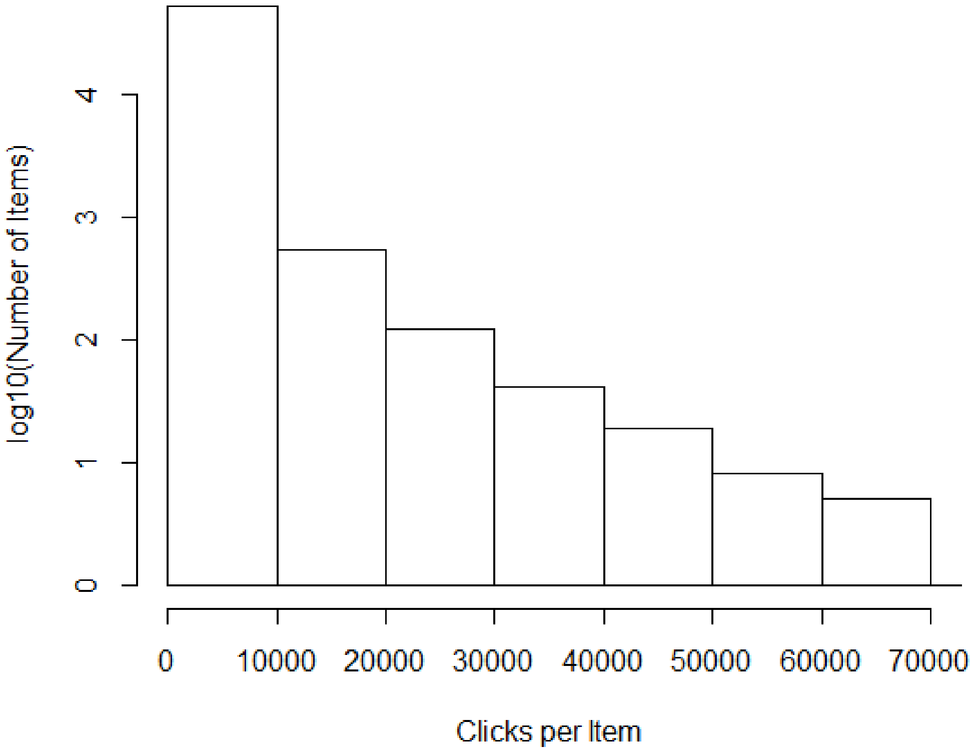


Figure 4.2: Frequency distribution of number of clicks per item in log scale at y-axis

in case we don't have state changes. We always assume that all stages are connected, so when we don't have transitions from one category to another, this creates problems in he calculation of likelihood. For these cases we made minimum bounds such as 1e-6 when we don't observe transition in order to overcome calculation problems. Also we put an upper bound for the times spend in each click to avoid error values, for example when a user forget his browser open and the time spent doesn't correspond to reality, refer to 4.3 and figure 4.8.

Further to data refining, only transitions to a different category are considered and a chain of distinct transitions is created. The final dataset contains session information with length from two page views to 48, and transitions among 289 categories. Also in the proposed model, web pages viewed by

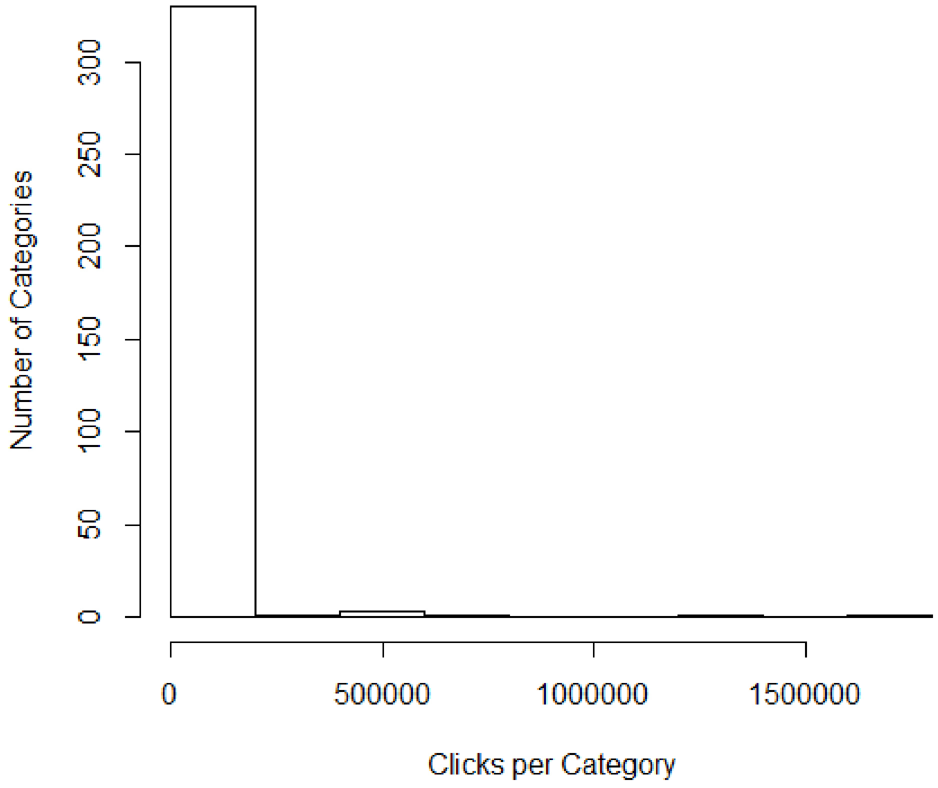


Figure 4.3: Frequency distribution of number of clicks per categories

	Length of paths	Time spent on a path	Time spent on a page
Minimum	1	0.0009999275	0.0009999275
Maximum	200	35100.69	6836.685
Median	2	166.897	63.098
Standard deviation	3.586181	767.7331	323.5106

Table 4.3: Data derived from 371984 calculated paths

the users will be the discrete states observable from the chain. The time between clicks, measured in seconds, will be the continuous component of the chain. These categories-web pages become the user-states in the model. The sequence of states for a particular visit by a user (session) becomes the

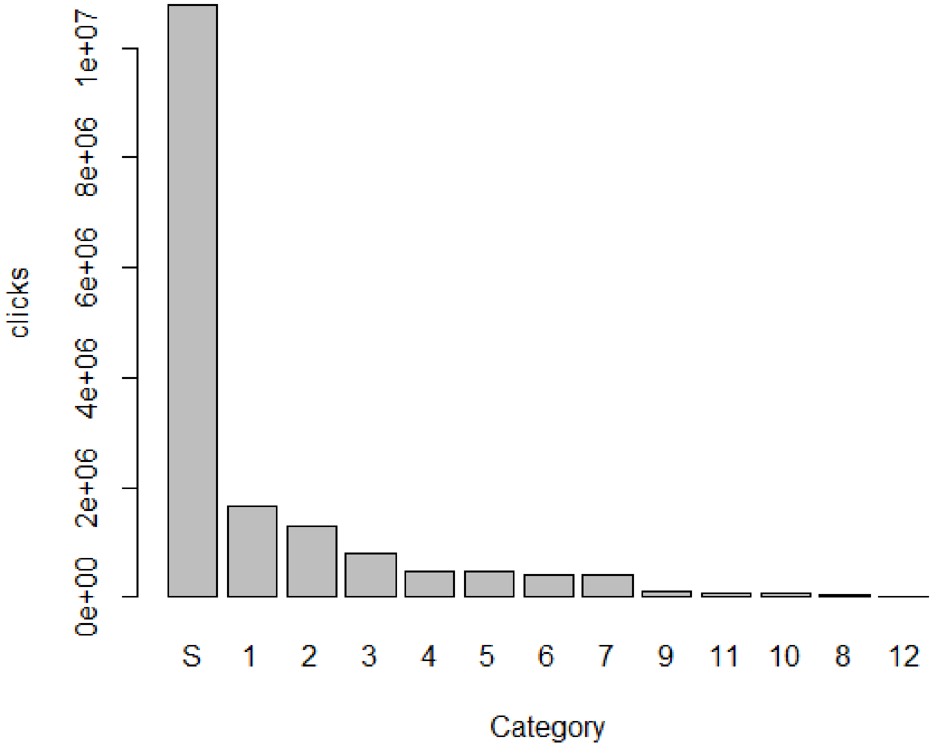


Figure 4.4: Most clicked categories

chain.

Listing 4.4: Sample of distinct/unique transitions (chain)

```
[[1]]”S” ”3” ”S” ”3” ”S”
[[2]]”S” ”2” ”S” ”2” ”S” ”5” ”S”
[[3]]”7” ”1” ”4” ”5” ”3” ”5” ”3” ”7”
[[4]]”4” ”S” ”2” ”4” ”S”
[[5]]”S” ”1” ”S” ”1”
[[6]]”1” ”S” ”1” ”S”
```

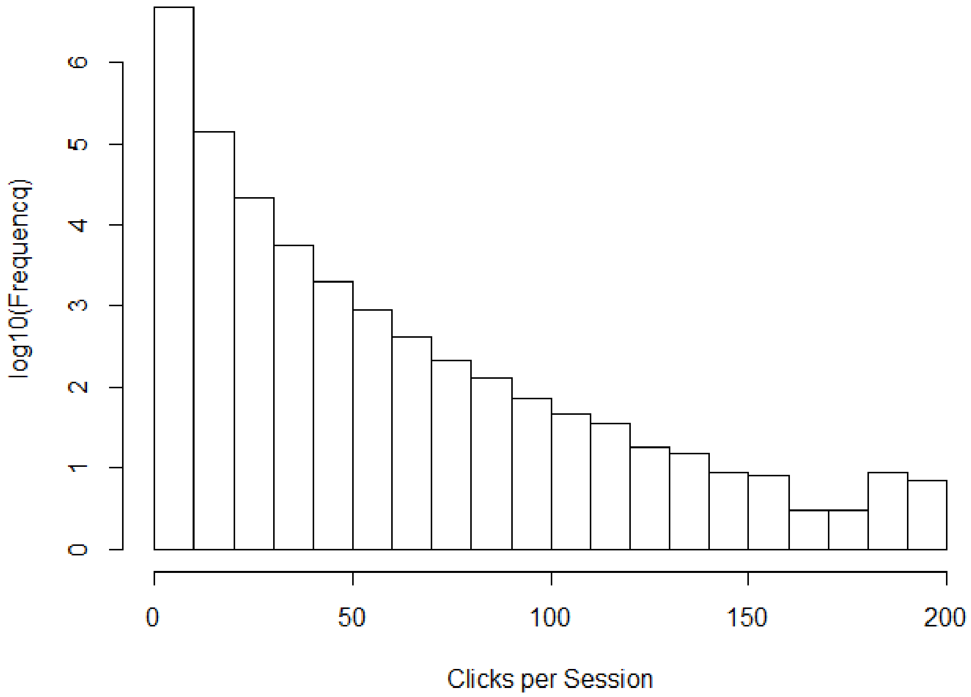


Figure 4.5: Complete cases: Frequency distribution of number of clicks per session in log scale at y-axis

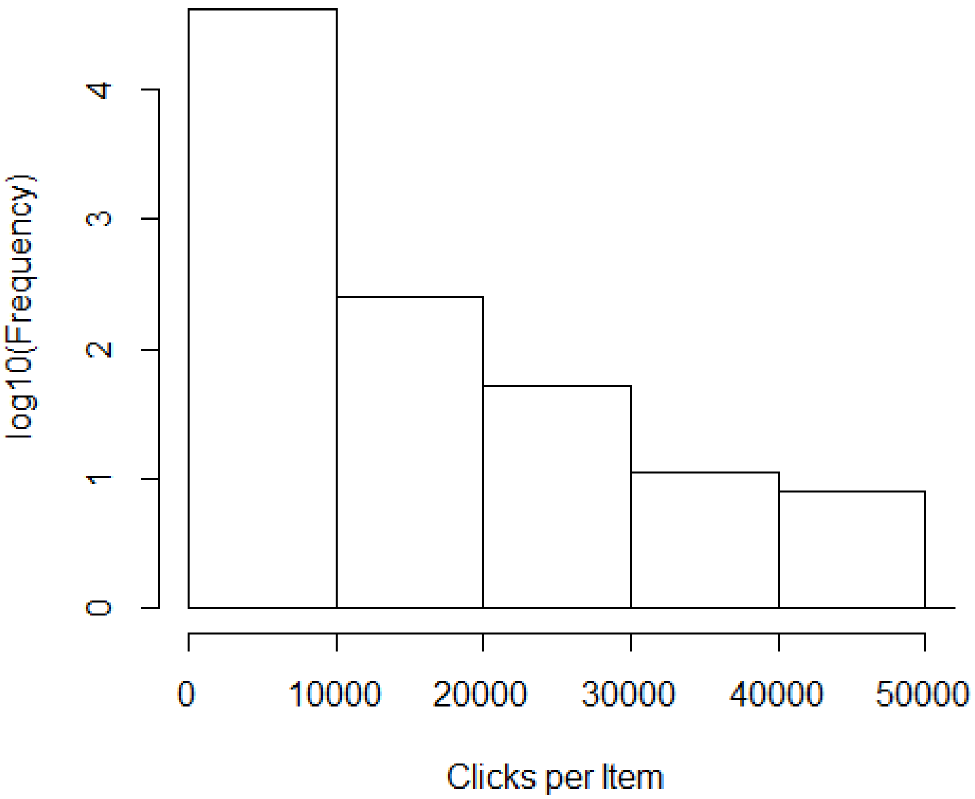


Figure 4.6: Complete cases:Frequency distribution of number of clicks per item in log scale at y-axis

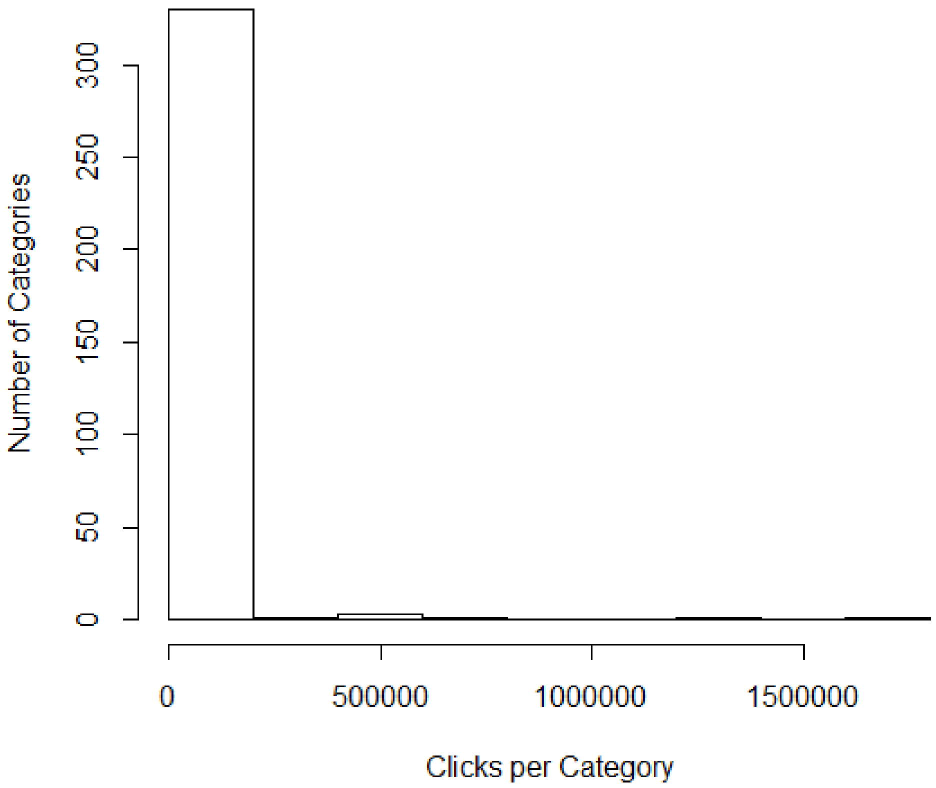


Figure 4.7: Complete cases:Frequency distribution of number of clicks per categories

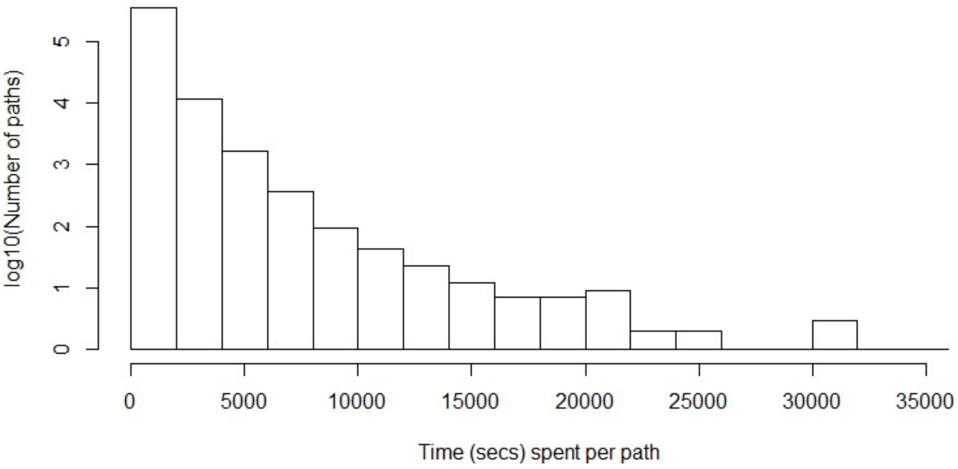


Figure 4.8: Histogram of time spent (in seconds) on any path in log scale at y-axis

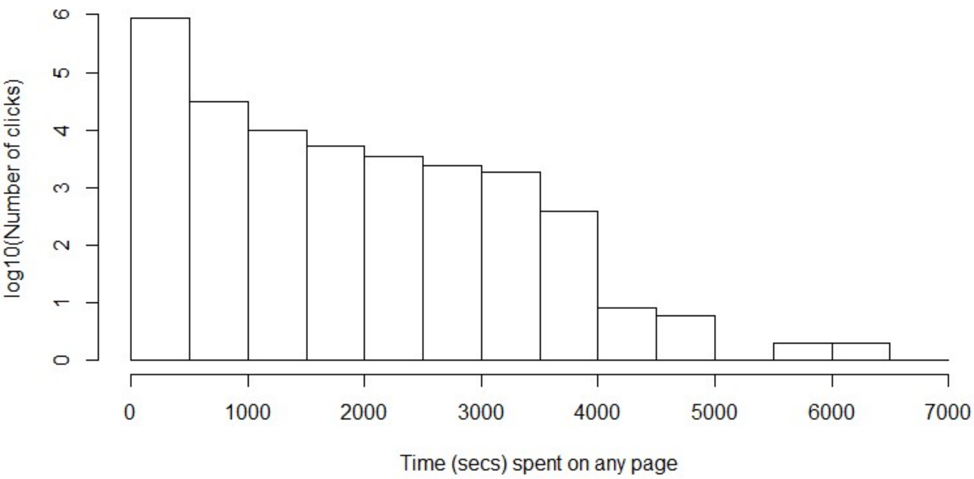


Figure 4.9: Histogram of time spent (in seconds) on any page in log scale at y-axis

4.3 Results

Using the clickstream data and the CTMC model developed by Albert et al. [1962], the Q matrix is calculated. This Q matrix is then used to calculate the probability of a user's next movement on the website including the amount of time which is spent in each state. The results show the time when the user's probability of going to the next page is at its highest before it starts a continuous decline. Using the Q matrix, the probability of jumping from state i to state j is calculated for all states over t from 1 to 289. $q(i, j)$ can be calculated by dividing the number of transitions from state i to state j by the total time spent in state i. We present the Q matrix for 289 categories so the matrix's dimension is 289×289.

$$\mathbf{Q}_g = \begin{pmatrix} -0.0007089872 & 1.92767 \cdot 10^{-6} & \cdots & 0.0004405962 \\ 0.0000391831 & -8.49271 \cdot 10^{-4} & \cdots & 0.0006743139 \\ \vdots & \vdots & \ddots & \vdots \\ 1.631046 \cdot 10^{-4} & 1.324882 \cdot 10^{-5} & \cdots & -0.0006791908 \end{pmatrix},$$

4.3.1 Finite mixture modelling with EM algorithm

Based in the methodology described in Section 3.4, we run the EM algorithm for finite mixtures with first-order CTMC components. Because EM may converge but not in a local maximum, a two-stage approach is incorporated, emEM (Biernacki et al. [2003]). The emEM runs multiple EM algorithms. Then the best solution is obtained and it used to initialize the final EM algorithm.

All transitions probabilities are greater than zero. To achieve that, we set the lowest probability value as a lower bound. This lower bound is assumed 10^{-6}.

Following with the EM algorithm, we fit the model into G groups, $G = 1, 2, 3, 4, 5$. To obtain our results we used the clickclust_EM command from r-package clickclust.cont (Gallaugher and McNicholas [2019]) with the following arguments :The maximum number of iterations is 50, The number of random starting values is 10, the tolerance for convergence is 1e-1. The r-package clickclust.cont provides an expectation maximization (EM) algorithm to fit a mixture of continuous time Markov models for use with clickstream. We choose the model with the lowest BIC . We present the resulting

BIC values in the Table 4.4.

Groups	G=1	G=2	G=3	G=4	G=5
BIC	31596.22	27961.94	28934.91	23629.64	25290.76

Table 4.4: BIC results for different number of groups.

G from 5 to 10 was tested but none of the models could be fitted.

Chapter 5

Conclusion

This study shows that a simple Continuous Time Markov Chain can be used to model the web users' behavior when browsing a website. A probability chart that displays how the users of that site behave over time can be developed. These probabilities only depend on the current page viewed by the users. This will provide a reference point in time when the probability of a user clicking on the next webpage will start a continuous decline.Because the probability does not have to be calculated for each individual user, this model is not as computing intensive. Comparing with the discrete time model one important aspect of the Discrete Time Markov Chain is the requirement of it to have discrete time between each state observation. This does not fit the nature of web browsing where users can "jump" from page to page at different time intervals. To try to fit the clickstream data into a Discrete Time Markov Chain would have forced the model to ignore any time information provided in the data. The results from Gallaugher and McNicholas [2018] confirm that taking into account the amount of time spent in each category allowed for the detection of groups of users. In contrast the discrete time model was unable to detect the group of users. Although this methodology assumes that the amount of time spend in a category follows exponential distribution and we checked the exponential property for our data this isn't always true in real time applications. The model is in this case time unit dependent. There are researches which consider a gamma distribution for the holding times but not for clickstream data. Also although BIC is a reliable inference criterion ,if the correct model is not on the list of the family models BIC tends to overestimate the size of clusters Biernaki et al. [1998].Contrary an integrated classification likelihood criterion(ICI) was developed. Other future research includes modeling it for a cross-websites application similar which is not considered in this model.Finally, this methodology was conceived for the purpose of analyzing clickstream data, but it has a more general application in every field that includes state transitions.

Appendix A

R Script

```
library(readr)
click<-read_csv("yoochoose-clicks.dat",
col_names = c("Session_ID","Timeclick","Item_ID","Category"),
col_types = cols(Category=col_character()),na = "0")
click_no_mising <- click[complete.cases(click), ]
i=1
sessions={}
path={}
times={}
sestimes={}
for(s in unique(click_no_mising$Session_ID)){
path[i] <- click_no_mising[click_no_mising$Session_ID == s, 4]
times[i] <- click_no_mising[click_no_mising$Session_ID == s, 2]
sestimes[[i]]<-c(as.character(s),diff(times[[i]], units="secs"))
sessions[[i]]<-c(as.character(s),path[[i]])
i=i+1
}

states = unique(unlist(path, use.names = FALSE))
for (a in sessions) {
cat(a,file="sessions.csv",sep=",",fill=FALSE,append=TRUE, "\n")
}
library(clickstream)
cls=readClickstreams(file="sessions.csv",sep=",",header=TRUE)
```

```
rotate0=function (x, n)
{
l = length(x)
n = n%%l
if (n == 0) {
return(x)
}
tmp = x[(l - n + 1):l]
x[(n + 1):l] = x[1:(l - n)]
x[1:n] = tmp
return(x)
}
getQ= function (i, clickstreamList)
{
clicks = clickstreamList
for (j in 1:i) {
clicks = rbind(clicks , "[[-]]")
}
clicks = unlist(clicks , use.names = F)
clicks2 = rotate0(clicks , -i)
dat = data.table(clicks , clicks2)
transition = as.data.frame(dcast.data.table(dat, clicks ~
clicks2 , fun.aggregate = length, value.var = "clicks2"))
transition = transition[, -1]
pos = which(names(transition) == "[[-]]")
rnames = names(transition)[-pos]
transition = transition[, -pos]
transition = transition[-pos, ]
names(transition) = rnames
rownames(transition) = rnames
return(list( transition = transition))
}

q1 = getQ(1, cls)
itrans=q1$transition

dif_times=lapply(times,diff,units="secs")
dif_times=lapply(dif_times,c,0)
```

```
nstat=length(names(itrans))
sum_times={}
sum_times= numeric(nstat)
library(parallel)
cr=detectCores()
library(doParallel)
cl <- makeCluster(cr-1)
registerDoParallel(cl)
s=lapply(sessions,tail,-1)
for(i in 1:nstat){
n=names(itrans)[i]
ind <- lapply(s, function(ch) grep(n, ch))
dtimes <- foreach (j=1:length(s), .combine = ’c’) %dopar% {
as.numeric(dif_times[[j]][ind[[j]]],units=”secs”)
}
sum_times[i] <- sum_times[i]+sum(dtimes,na.rm=TRUE)
}
names(sum_times)=names(itrans)

Q={}
for (i in 1:nstat){
if (sum_times[[i]] < 1e-6){
Q<- rbind(Q,itrans[i,]/1e-6)
}else{
Q<- rbind(Q,itrans[i,]/sum_times[[i]])
}
}
for (ij in 1:nstat){
Q[ij,ij]=-sum(Q[ij,][-ij])
}
Q<-matrix(unlist(Q),nrow = nstat, ncol = nstat)
```

Bibliography

C. C. Aggarwal, J. Han, J. Wang, and P. S. Yu. A framework for clustering evolving data streams. In *Proceedings of the 29th international conference on Very large data bases-Volume 29*, pages 81–92. VLDB Endowment, 2003.

A. Albert et al. Estimating the infinitesimal generator of a continuous time, finite state markov process. *The Annals of Mathematical Statistics*, 33(2): 727–753, 1962.

J. L. Andrews and P. D. Mcnicholas. Extending mixtures of multivariate t-factor analyzers. *Statistics and Computing*, 21(3):361–373, 2011.

C. Biernacki, G. Celeux, and G. Govaert. Choosing starting values for the em algorithm for getting the highest likelihood in multivariate gaussian mixture models. *Computational Statistics & Data Analysis*, 41(3-4):561–575, 2003.

C. Biernaki, G. Celeux, and G. Govaert. Assessing a mixture model for clustering with the integrated classification likelihood. Technical report, Technical Report, 1998.

D. Böhning and W. Seidel. Recent developments in mixture models. *Computational Statistics & Data Analysis*, 41(3-4):349–357, 2003.

D. Brodwin, D. O'Connell, and M. Valdmanis. Mining the clickstream. *Upside*, pages 101–106, 1995.

I. Cadez, D. Heckerman, C. Meek, P. Smyth, and S. White. Model-based clustering and visualization of navigation patterns on a web site. *Data mining and knowledge discovery*, 7(4):399–424, 2003.

P. H. Chatterjee. Dl and novak, tp. modeling the clickstream: Implications for web-based advertising efforts. *Marketing Science*, 22(4):520–541, 2003.

A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society: Series B (Methodological)*, 39(1):1–22, 1977.

M. Deshpande and G. Karypis. Selective markov models for predicting web page accesses. *ACM transactions on internet technology (TOIT)*, 4(2): 163–184, 2004.

L. Di Scala, L. La Rocca, and G. Consonni. A bayesian hierarchical model for the evaluation of a website. *Journal of Applied Statistics*, 31(1):15–27, 2004.

M. P. Gallaugher and P. D. McNicholas. Clustering and semi-supervised classification for clickstream data via mixture models. *arXiv preprint arXiv:1802.04849*, 2018.

M. P. Gallaugher and P. D. McNicholas. *ClickClustCont: Mixtures of Continuous Time Markov Models*, 2019. URL `https://CRAN.R-project.org/package=ClickClustCont`. R package version 0.1.7.

E. J. Johnson, W. W. Moe, P. S. Fader, S. Bellman, and G. L. Lohse. On the depth and dynamics of online search behavior. *Management science*, 50(3):299–308, 2004.

C. Keribin. Consistent estimation of the order of mixture models. *Sankhyā: The Indian Journal of Statistics, Series A*, pages 49–66, 2000.

J. Lee, M. Podlaseck, E. Schonberg, and R. Hoch. Visualization and analysis of clickstream data of online stores for understanding web merchandising. *Data Mining and Knowledge Discovery*, 5(1-2):59–84, 2001.

G. McLachlan and T. Krishnan. *The EM algorithm and extensions*, volume 382. John Wiley & Sons, 2007.

V. Melnykov. Model-based biclustering of clickstream data. *Computational Statistics & Data Analysis*, 93:31–45, 2016.

P. Metzner. *Transition path theory for Markov processes.* PhD thesis, Free University of Berlin, 2007.

B. Mobasher, H. Dai, T. Luo, and M. Nakagawa. Effective personalization based on association rule discovery from web usage data. In *Proceedings of the 3rd international workshop on Web information and data management*, pages 9–15. ACM, 2001.

A. L. Montgomery and C. Faloutsos. Identifying web browsing trends and patterns. *Computer*, 34(7):94–95, 2001.

A. L. Montgomery, S. Li, K. Srinivasan, and J. C. Liechty. Modeling online browsing and path analysis using clickstream data. *Marketing science*, 23 (4):579–595, 2004.

Y.-H. Park and P. S. Fader. Modeling browsing behavior at multiple websites. *Marketing Science*, 23(3):280–303, 2004.

J. Pitkow and P. Pirolli. Mininglongestrepeatin g subsequencestopredict worldwidewebsurfing. In *Proc. UsENIX symp. on Internet Technologies and systems*, page 1, 1999.

M. Scholz et al. R package clickstream: analyzing clickstream data with markov chains. *Journal of Statistical Software*, 74(4):1–17, 2016.

G. Schwarz et al. Estimating the dimension of a model. *The annals of statistics*, 6(2):461–464, 1978.

S. L. Scott and I.-H. Hann. A nested hidden markov model for internet browsing behavior. *Marshall School of Business*, 2006.

P. Smyth. Clustering sequences with hidden markov models. In *Advances in neural information processing systems*, pages 648–654, 1997.

J. Wei, Z. Shen, N. Sundaresan, and K.-L. Ma. Visual cluster exploration of web clickstream data. In *2012 IEEE Conference on Visual Analytics Science and Technology (VAST)*, pages 3–12. IEEE, 2012.