Modelling Visit Similarity Using Click-Stream Data: A Supervised Approach

Deepak Pai[1], Abhijit Sharang[2],
Meghanath Macha Yadagiri[1], and Shradha Agrawal[1]

[1] Adobe Research
{deepak.pai,mmacha,shraagra}@adobe.com
[2] Indian Institute of Technology, Kanpur
abhisg@iitk.ac.in

Abstract. Identifying and targeting visitors on e-commerce website with personalized content in real-time is extremely important to marketers. Although such targeting exists today, it is based on demographic attributes of the visitors. We show that dynamic visitor attributes extracted from their click-stream provide much better predictive capabilities of visitor intent. In this work, we propose a mechanism for identifying similar visitor sessions on a website based on their click-streams. Novel techniques for extracting features from visitor clicks are employed. Large margin nearest neighbour (LMNN) algorithm is used to learn a similarity metric between any two sessions. Further the sessions are classified into purchasers and non-purchasers using k-nearest neighbour (kNN) classification. Experimental results showing significant improvements over baseline algorithms based on Hidden Markov Model(HMM), support vector machine (SVM) and random forest are presented on two large real-world data sets.

Keywords: click-stream, metric learning, kNN classification, e-Commerce.

1 Introduction

Organizations are embracing online platforms to increase their reach and reduce operational cost of brick and mortar stores. Of late even software companies are moving towards online purchase and subscription models. Although web-visitors form a significant chunk of customer base for these retailers, the conversion rates are not very encouraging yet. E-commerce sites typically see conversions in the range of 1-3 % [14], which is much less compared to conversions in offline world. Hence organizations are interested in analysing online visits to understand low conversion rates and influence customers to make purchases. Marketing techniques like web-page personalization, targeted campaigns, promotional offers etc. are routinely adopted to persuade visitors. While these techniques increase the conversions, they incur significant marketing budget. Marketing to visitors who do not convert, amounts to wasted marketing dollars. Hence marketers would

B. Benatallah et al. (Eds.): WISE 2014, Part I, LNCS 8786, pp. 135–145, 2014.

like to know appropriate visitors, who would respond positively to a targeted campaign. We propose an approach to identify this target group for any marketing campaign, based on visitor click-streams.

Click-stream refers to the series of URLs visited by the customer. The information is available from logs generated by the servers or javascript tags embedded into the web-pages. Besides URL information, meta-data such as geographic location, purchase information, page categories, campaign information, referrer pages, etc. are also collected. This information is used to gain insight into the user intent, and subsequently predict potential leads, churn or any such desired metric. Clustering of visitor sessions by appropriate features generates market segments for targeted campaigns. Look-a-like modelling is another technique used by marketers to analyse visits. Applications of such modelling techniques include fraud detection, web page personalization, offline reporting, market segmentation, offline targeting, real-time online targeting etc.

At the heart of modelling techniques such as look-a-like modelling, clustering etc. lies the metric for identifying similarity of a visit pair. The features and their combination used for deriving such a metric has a large influence on the outcome of these techniques. Our primary contribution in the proposed work lies in building relevant features from click-stream data and then devising an appropriate metric for identifying a similarity score among visits. The contributions in this paper are multi-fold. Firstly we define the features for clustering among the multitude of features available from click-stream. Secondly we learn the feature weights using a supervised learning algorithm based on LMNN. Finally we show that the metric thus learnt is actually useful, by classifying the data using a kNN classifier. The superiority of the feature engineering and classification scheme is shown by comparing it with state-of-the-art algorithms like HMM, SVM and random forests.

The rest of the paper is organized as follow. We first describe related work in the areas of click-stream analytics. Next we define our problem in formal terms and our proposed solution approach. In the subsequent sections, we describe our experimental set-up, results and discuss the relevance of the results. Finally we conclude and discuss possible future enhancements.

2 Related Work

Classification of visitors based on click-streams is reasonably well explored area. Markov chains and Hidden Markov models (HMMs) are commonly applied to model click data. Scott and Hann [15] classify visitor sessions into one of predefined categories, namely decisive shoppers, deliberators, and never buyers, with a nested HMM. The higher level HMM models visits across sessions, while the nested lower level HMM captures clicks in a given session. Montgomery et. al. [11] have modelled browsing behaviour as dynamic multinomial probit models. Further they propose to capture previous clicks into higher order Markov models. HMMs are also used by Ypma et. al. [20] to categorize web-pages. Further they clustered users based on the observed click-streams. Sismeiro and Bucklin [3]

propose a Tobit model capturing browser behaviour. Empirical results showing variation in browsing intent with depth of visit and number of repeat visits are presented.

Recent work has explored visitor clustering from their on-site behaviour. Li, Tian, and Xing [7] compute similarity of visitor sessions based on web-page attributes using Euclidean distance measure. The similarity is further used to cluster visits using k-medoids clustering algorithm. Petridou et. al. [13] cluster visitors considering the visit behaviour and time of viewing. Moe et. al. [10] categorize users into directed buyers, hedonic browsers, search deliberators, and knowledge seekers. Visits are clustered into one of these groups based on the browsing characteristics across various web-page categories.

Prior work has concentrated on modelling click-streams as first or second order Markov chains [11]. Intuitively it feels more likely that visitor choices are dependent on their entire history of clicks in the current as well as certain previous sessions. However higher order Markov models are more complicated to model, especially in BigData scenario. HMMs address this problem to a certain extent, but prior work of modelling click-streams with HMMs seem limited. Moreover, HMMs show significant bias towards the majority class, which limits their application in case of data with class imbalance, such as ours. Additionally, capturing metrics like visit counts, repeat visits, and visit duration into HMM observation sequence is non-trivial and noticeably previous work on HMM and Markov chains have not captured this information. Moreover,our experiments showed that the performance of HMMs was poor and comparable only to random oracle. We observe from our approach that features are significant indicators of visitor intent. The available clustering techniques consider such metrics, however each dimension is given equal weight. Mahalanobis distance [8] better captures the distance between two points, in a transformed dimension. Blitzer, Weinberger, and Saul [2] propose a method for learning Mahalanobis distance from labelled data points as a supervised learning algorithm. The algorithm is optimized to improve kNN classification results. Such an algorithm has been effectively used in literature [9] for variety of applications. To the best of our knowledge, there has been no previous work in exploring LMNN algorithm with Mahalanobis distance for classifying or clustering visitor sessions. In this work we present our results of such an exploration and show improvements over baseline algorithms, empirically on two large real-world data-sets.

3 Problem Definition

We formally define our problem in this section. We start with an intuition to the proposed solution.

We model the problem at a session level rather than a visitor level, since different sessions of the same user are not necessarily the same. Online purchase cycle of users interested in making a purchase could be broken down into logical stages. Such techniques have been used effectively by [10,16]. Initial user sessions are geared towards deliberating on the product of purchase since the user is more

interested in looking at various available options to choose from. Hence the visits are concentrated on informative pages, which provide data to make a decision. At the end of this phase the user has a narrowed down a list of items, among which, he would make a purchase. Later stages involve repeated visits to items in the interest set, ending with a purchase. However, most visitors on the website arrive only to browse and update their knowledge about certain products without having an intent of purchasing. Moreover, in case of e-commerce websites, visitor interest changes rapidly. A visitor interested in purchasing a gaming console today, might be interested in purchasing books, few months later. Combining information across visitor sessions, therefore results in a heterogeneous mixture of interests and does not yield good results. Intuitively identifying visitors in deliberation phase and influencing them would be more useful than targeting users with non-purchase intent or users who have already made a decision. Hence our proposed work concentrates on modelling visits at a session level rather than visitor level.

Let $\mathbb{S} = \{S_1, S_2, ..., S_n\}$ be the set of user sessions or visits.

Let $S_i = \{C_i^1, C_i^2, ..., C_i^m\}$ be the click-stream with C_i^k representing k^{th} click of i^{th} session. C_i^k is a vector of features for every click. The feature vector is formed from the information derived from meta-data corresponding to the clicks.

Let $C_i^k = \{f_{ik}^1, f_{ik}^2, ..., f_{ik}^p\}$ be the feature values corresponding to k^{th} click of i^{th} session. Let p_i be the binary label associated with session S_i with 1 representing a purchase.

We want to compute a similarity metric Sim_{ij} for any given pair of sessions S_i and S_j, based on $\mathbb{F}$. Hypothetically similar sessions will share the same label as compared to dissimilar sessions.

Given the similarity matrix $Sim = (sim_{ij})_{i,j=1..n}$, we need to label points in the test-set, based on the labels of training-set in their neighbourhood. Let $S_{i,test}$ be any point in the test-set, that needs to be labelled.

Let $S_{i,train} = \{S_{i,train}^1, S_{i,train}^2, ..., S_{i,train}^k\}$ be the data points in the training-set, which are in the neighbourhood of $S_{i,test}$. The label of $S_{i,test}$ is decided by majority voting of elements of $S_{i,train}$.

4 Approach

Our approach broadly consists of categorising the URLs, sessionizing URLs into user sessions, selecting features that best capture the visitor behaviour on the website in a given session, learning a metric to find similar sessions and finally assigning labels to sessions based on the learnt metric. We discuss the details in the subsequent sections.

4.1 Categorisation

A reasonably large website consists of ten or hundred thousand URLs and thousands of unique URLs. To consider each of these URLs as a separate dimension would lead to very sparse matrix on which learning algorithms do not perform

well. Moreover scaling the algorithms to such large feature dimensions is not trivial. To deal with this problem [15,11,10] categorised the URLs into pre-defined categories. Following a similar approach we initially categorised the URLs into pre-defined categories.

Let $U = \{u_1, u_2, ..., u_n\}$ be the unique URLs and $Cat = \{cat_1, cat_2, ..., cat_p\}$ be the pre-defined categories. We define a mapping function $\mathbb{M}$, which maps U to Cat.

$\mathbb{M}(u_i) = cat_j \ \forall \ u_i \in U \mid cat_j \in Cat$

Henceforth our analysis would be on the transformed space Cat.

4.2 Sessionization

From the raw click-stream data, we need to form logical sessions corresponding to user visit. The sessionization is done based on thirty minutes of inactivity. For every user, from the start of his session, till the point of inaction for thirty minutes, all URLs are grouped to form a single visitor session.

4.3 Feature Selection

We first compute the latent feature–time spent on the category. This is computed by considering the difference in time-stamps of subsequent clicks. After analysing all available features / meta-data for each click, visit count and the visit duration of the category are found to be the most indicative features. The categories defined in the previous section $Cat(Metric, S_i)$ form our features, where $Metric$ is either visit count or visit duration. First we compute the raw feature values for each session from the click-stream. Visit count values for a category cat_j in any given session S_i is the frequency of visits to the category in the session.

$$Cat(VisitCount, S_i) = \{cat_i^1, cat_i^2, ..., cat_i^p\} \text{ where } cat_i^j = Frequency(cat_j) \text{ in } S_i$$

$$cat_i^j = \sum_{j=1}^{p} v_i^j \mid v_i^j = \begin{cases} 1 & \text{if } \mathbb{M}(f_{ik}^l) = cat_j \ \forall \ C_i \in S_i \text{ and } f_{ik}^l = URL. \\ 0 & \text{otherwise} \end{cases}$$

Similarly, we compute raw feature values for the metric, visit duration as summation of visit durations to the categories over all clicks in that session and normalize the values by computing the term frequency-inverse document frequency (tf-idf) score from the click-stream. tf-idf [6,1,19] is widely used in text processing, where individual words or n-grams form the feature vector. In our scenario, categories, sessions, and set of all sessions are analogous to words / n-grams, documents, and document corpus respectively. The need for normalization is intuitive. People on an average might spend more time on product-1 as compared to product-2 or an individual user-1 might take more time looking through product reviews than user-2. Normalizing the features will remove such biases.

4.4 Metric Learning

As the next step, we learn weights for each feature in our feature vector $\mathbb{F}$. The weights are then used to learn a metric of distance between any two data points. One of the most common metrics is the **Euclidean metric** which assigns equal weights to all features and is hence not optimal. This problem is tackled by taking the Euclidean distance in a transformed space where the variance of the dimensions in the feature vector is taken into account. The transformed space can be achieved by introducing a matrix L into the distance function.

$$\mathbb{D}(\boldsymbol{x_i}, \boldsymbol{x_j}) = (\boldsymbol{x_i} - \boldsymbol{x_j})^T L^T L(\boldsymbol{x_i} - \boldsymbol{x_j})$$

The above equation transforms to

$$\mathbb{D}(\boldsymbol{x_i}, \boldsymbol{x_j}) = (L\boldsymbol{x_i} - L\boldsymbol{x_j})^T (L\boldsymbol{x_i} - L\boldsymbol{x_j})$$

which is the Euclidean distance in a transformed space. The metric is called Mahalanobis metric. Blitzer, Weinberger, and Saul [2] propose learning a Mahalanobis distance metric for kNN classification from labelled examples. In the subsequent sections we briefly discuss about the algorithm and the proposed improvements.

Large Margin Nearest Neighbour. In this section we briefly describe the LMNN approach. The algorithm is a measure to improve the accuracy of the kNN classification algorithm. The goal of the algorithm is to ensure that all k neighbours for a test point belong to the same class. This is achieved by learning a positive semi-definite matrix M, which transforms the original vector space to transformed space. The matrix $M = L^T L$ is arrived at by solving a convex optimisation problem with additional constraints. Thus the problem is reformulated to be semi-definitive program(SDP) [17]. The distance function is hence formulated as

$$\mathbb{D}(\boldsymbol{x_i}, \boldsymbol{x_j}) = (\boldsymbol{x_i} - \boldsymbol{x_j})^T M(\boldsymbol{x_i} - \boldsymbol{x_j})$$

Thus the SDP is formulated to minimize the following cost function:

$$\sum_{ij} \eta_{ij} \mathbb{D}(\boldsymbol{x_i}, \boldsymbol{x_j}) + \mathrm{c} \sum_{i,j,k} \eta_{ij}(1 - p_{ik})[1 + \mathbb{D}(\boldsymbol{x_i}, \boldsymbol{x_j}) - \mathbb{D}(\boldsymbol{x_i}, \boldsymbol{x_k})], \tag{1}$$

where $p_{ij} \in \{0, 1\}$ is used to indicate whether labels p_i and p_j match or not. $\eta_{ij} \in \{0, 1\}$ is used to indicate whether $\boldsymbol{x_i}$ and $\boldsymbol{x_j}$ are target neighbours or not. For every data point x_i, set of k target neighbours are initially identified. The target set, η_{ij} remains unchanged during learning. c is a positive constant generally set by cross validation.

Additional constraints are

$$\begin{gathered} \mathbb{D}(\boldsymbol{x_i}, \boldsymbol{x_l}) - \mathbb{D}(\boldsymbol{x_i}, \boldsymbol{x_j}) \geq \mathbb{D}(\boldsymbol{x_i}, \boldsymbol{x_j}) - \mathbb{D}(\boldsymbol{x_i}, \boldsymbol{x_l}) \\ 1 + \mathbb{D}(\boldsymbol{x_i}, \boldsymbol{x_j}) - \mathbb{D}(\boldsymbol{x_i}, \boldsymbol{x_l}) \geq 0 \\ M \succeq 0 \end{gathered}$$

The first term in eq. (1) penalises large distances between each data point and its target neighbours, while the second term penalizes small distances between each data point and all other data points with different labels. The final constraint restricting M to be positive semi-definitive.

4.5 kNN Classification

Finally, we apply a straight forward kNN classifier with majority voting to label the test data.

5 Experimental Set-Up

In this section we describe our experimental set-up, data-set used and the results of our explorations. We evaluate our approach on two large real-world datasets: the first from a software company that sells products and subscriptions online, and the second from an e-Commerce retailer specializing in the sale of hiking and camping products. Also, we show that the nature of our data-sets is quite generic and hence the proposed algorithms could be readily applied to any other data-set.

5.1 Data Description

Our dataset consisted of a single day worth of click-streams for each company.

Data-Set I. The first data-set had $19,737,907$ sessions. The number of purchases were 1444 giving us a conversion rate of 0.007%. The average click length was 2.7. Average time spent by users in a session was 186 seconds.

Data-Set II. The second data-set had $211,577$ sessions. The number of purchases were 3459 giving us a conversion rate of 1.6%. The average click length was 11. Average time spent by users in a session was 298 seconds.

5.2 Categories

As described in the earlier section, we categorize URLs into appropriate categories. The categories are predefined based on their relevance to the data-sets. In case of our first data-set the categories used were Cat = {Home, Product, Account, Info, Trial, ShopCart, Order, Help}. Categories Cat = {Home, Product, Account, Category, Info, ShopCart, Search, Order}, were formed in case of the second data-set.

6 Results and Discussion

In this section we present the results of our analysis. Further we discuss the results and ways this information could be used by marketers to perform online or offline targeting of customers. We compare the results of our algorithm, against results from HMM, SVM and random forest, that are used as the baseline algorithms.

6.1 Baseline Algorithms

We compare our results against multiple baseline algorithms. Below we briefly describe the set-up of each algorithm.

HMM. We build HMM model of the click data on the lines of [11]. The webpages are categorized as described earlier. The categories form the set of observations. We model the HMM to have two hidden states indicative of purchasers and non-purchasers. Initial state probabilities, state transition probabilities and emission probabilities are computed as prior from the training data. Viterbi and BaumWelsh algorithms are used for training purposes and compute the HMM parameters. The model thus built is used to predict the visitor state from the observation sequences in test data.

SVM. We train a SVM classifier with linear, polynomial, radial and sigmoid kernels. Visitor information such as country, referrer search engine, demographic region, number of clicks and total time spent on the website are used as features.

Random Forest. Here, we use the same features as in SVM to train a random forest classifier. The model thus built is used to predict labels for the test data.

6.2 Validation

We are interested in classifying user sessions, as early as possible in the session. We split the click-streams vertically into $25, 50$, and 75 % of their total length. As we are interested in identifying visitors who are likely to make a purchase, these visitors form our positive class. All metrics that are reported are for positive (minority) class. We divide our data into training and test data with a 80-20 split. LMNN algorithm is trained to arrive at the covariance matrix $\mathbb{M}$. The Mahalanobis distance between any two sessions Sim_{ij}, which forms our distance metric is computed as described earlier. We compute the similarity metric for sessions in the test data to find k nearest neighbours. We experimented with values of k ranging from 5 to 15 and selected 13 as an appropriate value of k based on empirical evidence. Labels for each point in the test-set are then computed based on majority voting of training data points in the neighbourhood. We looked at the variation in precision and recall over different values of k for

25% split of the data and observed that the best results are obtained for $k = 13$. While best precision values are observed at $k = 13$, recall values remain constant. The precision recall values for $k = 13$ and 25% split of click-stream are shown in Table 1. Intuitively kNN results in best classification when only the most relevant neighbours are used for classification, with any increase or decrease in the number of neighbours resulting in data points from other classes dominating the classification, especially with large class imbalance. We combine the precision and recall values into F1 score, to better compare with the performance of baseline algorithms. F1 values of the proposed algorithm in comparison with the baseline algorithms are also shown in the same table. Similarly P-R values and F1 scores for other data splits (50 and 75) are shown in Tables 2- 3 respectively. Value of w_1 and w_0 are set by cross validation.

6.3 Discussion

We observe that our algorithm performs significantly better than the baseline algorithms. Results show higher precision, recall, and F1 values for our algorithm as compared to the baseline algorithms. SVM classifier always predicted the majority class resulting in precision, recall and F1 score being all 0. Hence we have not included the results of SVM in the result tables. HMM and random forests are seen to perform extremely poorly. This is primarily due to the huge class imbalance in the data. HMM for instance has a near cent percent start probability of visitor being a non-purchaser. The transition model built is diagonal heavy indicating that the visitors are most likely to remain in the category that they are in. Eventually the model predicts the visitor being a non-purchaser most of the time. A similar behaviour is observed in case of random forest as well. Not surprisingly the precision and recall values are very low for both baseline algorithms. Variations of LMNN with Euclidian distance and Mahalanobis distance show comparable performance. The results are consistent across various splits of the click-streams. Moreover it is observed that the precision and recall improve as we observe more data i.e. as we move from 25 to 50% or 50 to 75% data split. Comparable results are obtained across data-sets from significantly different domains, indicating the generality of the proposed approach. We observe that visitors not having a purchase intent spend more time on help related pages like blogs and forums. This is explained by the fact that these visitors have already purchased the software and are currently more interested in getting it working.Not surprisingly, visitors having a purchase intent spend time on shopping cart related pages, adding / removing items and modifying quantity of items to purchase. Visits resulting in a purchase involve lesser number of product views but larger amount of time spent on product pages as compared to visits that do not result in a purchase. This is consistent with our hypothesis that visitors in the deliberation phase look at variety of products before making a purchase decision. The actual purchase happens in the later sessions which are characteristic of decisive buyers, who look at few products that they are most interested in and spend large amount of time gathering details of these products. Visits and time spent on home page confirms this behaviour, with buyers

spending much lesser time on home page. Initial visitor sessions are seen to be starting from home page, whereas decisive buyers are more likely to land on the product page they are interested in.

7 Conclusion and Future work

An algorithm to classify online visitor sessions based on their click-streams is presented. The classification is based on kNN algorithm based on a novel similarity metric. The similarity metric is computed as a Mahalanobis distance between two visitor sessions. The feature weights or covariance matrix is learnt from LMNN algorithm. Our feature engineering in combination with the classification scheme outperform baseline algorithms, based on HMM, SVM and random forest, as shown from empirical evidence with two large real-world data-sets. The consistency of the algorithm over varied data-sets, shows the generality of our approach. Although the current work focuses on identifying potential purchases, the similarity metric is equally applicable to other applications like churn prediction, look alike modelling, visitor segmentation, etc. that will be explored in future work.

Table 1. Precision Recall values for k=13 and 25 % split

	Data-set I			Data-set II		
Algorithm	Precision	Recall	F1	Precision	Recall	F1
HMM	0.026	0.264	0.047	0.026	0.27	0.047
Random Forest	0.003	0.007	0.004	0.083	0.027	0.041
LMNN-Euclidean	0.804	0.153	0.256	0.785	0.342	0.477
LMNN-Mahalanobis	0.807	0.156	0.261	0.788	0.342	0.477

Table 2. Precision Recall values for k=13 and 50 % split

	Data-set I			Data-set II		
Algorithm	Precision	Recall	F1	Precision	Recall	F1
HMM	0.027	0.26	0.049	0.026	0.258	0.047
Random Forest	0.025	0.029	0.027	0.138	0.023	0.039
LMNN-Euclidean	0.795	0.461	0.584	0.857	0.490	0.624
LMNN-Mahalanobis	0.801	0.451	0.577	0.868	0.489	0.625

Table 3. Precision Recall values for k=13 and 75 % split

	Data-set I			Data-set II		
Algorithm	Precision	Recall	F1	Precision	Recall	F1
HMM	0.023	0.184	0.041	0.024	0.197	0.043
Random Forest	0.043	0.009	0.015	0.141	0.016	0.029
LMNN-Euclidean	0.795	0.617	0.695	0.860	0.609	0.713
LMNN-Mahalanobis	0.802	0.661	0.724	0.860	0.617	0.718

References

1. Aizawa, A.: An information-theoretic perspective of tf–idf measures. Information Processing & Management 39(1), 45–65 (2003)
2. Blitzer, J., Weinberger, K.Q., Saul, L.K.: Distance metric learning for large margin nearest neighbor classification. In: Advances in Neural Information Processing Systems, pp. 1473–1480 (2005)
3. Bucklin, R.E., Sismeiro, C.: A model of web site browsing behavior estimated on clickstream data. Journal of Marketing Research, 249–267 (2003)
4. Cadez, I., Heckerman, D., Meek, C., Smyth, P., White, S.: Visualization of navigation patterns on a web site using model-based clustering. In: Proceedings of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 280–284. ACM (2000)
5. Faloutsos, M., Faloutsos, P., Faloutsos, C.: On power-law relationships of the internet topology. In: ACM SIGCOMM Computer Communication Review, vol. 29, pp. 251–262. ACM (1999)
6. Joachims, T.: A probabilistic analysis of the rocchio algorithm with tfidf for text categorization. Technical report, DTIC Document (1996)
7. Li, J., Tian, H., Xing, D.: Clustering user session data for web applications test. Journal of Computational Information Systems 7(9), 3174–3181 (2011)
8. Mahalanobis, P.C.: On the generalized distance in statistics. Proceedings of the National Institute of Sciences (Calcutta) 2, 49–55 (1936)
9. Mensink, T., Verbeek, J., Perronnin, F., Csurka, G.: Metric learning for large scale image classification: Generalizing to new classes at near-zero cost. In: Fitzgibbon, A., Lazebnik, S., Perona, P., Sato, Y., Schmid, C. (eds.) ECCV 2012, Part II. LNCS, vol. 7573, pp. 488–501. Springer, Heidelberg (2012)
10. Moe, W.W.: Buying, searching, or browsing: Differentiating between online shoppers using in-store navigational clickstream. Journal of Consumer Psychology 13(1), 29–39 (2003)
11. Montgomery, A.L., Li, S., Srinivasan, K., Liechty, J.C.: Modeling online browsing and path analysis using clickstream data. Marketing Science (2004)
12. Newman, M.E.: Power laws, pareto distributions and zipf's law. Contemporary Physics 46(5), 323–351 (2005)
13. Petridou, S.G., Koutsonikola, V.A., Vakali, A.I., Papadimitriou, G.I.: Time-aware web users' clustering. IEEE Transactions on Knowledge and Data Engineering 20(5), 653–667 (2008)
14. Poggi, N., Carrera, D., Gavalda, R., Ayguadé, E., Torres, J.: A methodology for the evaluation of high response time on e-commerce users and sales
15. Scott, S.L., Hann, I.-H.: A nested hidden markov model for internet browsing behavior (2006)
16. Sismeiro, C., Bucklin, R.E.: Modeling purchase behavior at an e-commerce web site: a task-completion approach. Journal of Marketing Research (2004)
17. Vandenberghe, L., Boyd, S.: Semidefinite programming. SIAM Review (1996)
18. Weinberger, K.Q., Saul, L.K.: Distance metric learning for large margin nearest neighbor classification. J. Mach. Learn. Res. 10, 207–244 (2009)
19. Wu, H.C., Luk, R.W.P., Wong, K.F., Kwok, K.L.: Interpreting tf-idf term weights as making relevance decisions. ACM Transactions on Information Systems (TOIS) 26(3), 13 (2008)
20. Ypma, A., Ypma, E., Heskes, T.: Categorization of web pages and user clustering with mixtures of hidden markov models (2002)