Neural Computing and Applications
https://doi.org/10.1007/s00521-018-3523-0

ORIGINAL ARTICLE

Real-time prediction of online shoppers' purchasing intention using multilayer perceptron and LSTM recurrent neural networks

C. Okan Sakar[1] · S. Olcay Polat[2] · Mete Katircioglu[1] · Yomi Kastro[3]

Received: 18 July 2017 / Accepted: 4 May 2018

Abstract
In this paper, we propose a real-time online shopper behavior analysis system consisting of two modules which simultaneously predicts the visitor's shopping intent and Web site abandonment likelihood. In the first module, we predict the purchasing intention of the visitor using aggregated pageview data kept track during the visit along with some session and user information. The extracted features are fed to random forest (RF), support vector machines (SVMs), and multilayer perceptron (MLP) classifiers as input. We use oversampling and feature selection preprocessing steps to improve the performance and scalability of the classifiers. The results show that MLP that is calculated using resilient backpropagation algorithm with weight backtracking produces significantly higher accuracy and F1 Score than RF and SVM. Another finding is that although clickstream data obtained from the navigation path followed during the online visit convey important information about the purchasing intention of the visitor, combining them with session information-based features that possess unique information about the purchasing interest improves the success rate of the system. In the second module, using only sequential clickstream data, we train a long short-term memory-based recurrent neural network that generates a sigmoid output showing the probability estimate of visitor's intention to leave the site without finalizing the transaction in a prediction horizon. The modules are used together to determine the visitors which have purchasing intention but are likely to leave the site in the prediction horizon and take actions accordingly to improve the Web site abandonment and purchase conversion rates. Our findings support the feasibility of accurate and scalable purchasing intention prediction for virtual shopping environment using clickstream and session information data.

Keywords Online shopper behavior · Shopping cart abandonment · Clickstream data · Deep learning

1 Introduction

The increase in e-commerce usage over the past few years has created potential in the market, but the fact that the conversion rates have not increased at the same rate leads to the need for solutions that present customized promotions to the online shoppers [1–3]. In physical retailing, a salesperson can offer a range of customized alternatives to shoppers based on the experience he or she has gained over time. This experience has an important influence on the effective use of time, purchase conversion rates, and sales figures [4]. Many e-commerce and information technology companies invest in early detection and behavioral prediction systems which imitate the behavior of a salesperson in virtual shopping environment [2, 5, 6]. In parallel with these efforts, some academic studies addressing the problem from different perspectives using machine learning

✉ C. Okan Sakar
okan.sakar@eng.bau.edu.tr

S. Olcay Polat
polat_suleymanolcay@columbusstate.edu

Mete Katircioglu
alpaslan.katircioglu@stu.bahcesehir.edu.tr

Yomi Kastro
yomi.kastro@inveon.com.tr

[1] Department of Computer Engineering, Faculty of Engineering and Natural Sciences, Bahcesehir University, 34349 Besiktas, Istanbul, Turkey

[2] TSYS School of Computer Science, Columbus State University, Columbus, USA

[3] Inveon Information Technologies Consultancy and Trade, 34335 Istanbul, Turkey

Published online: 09 May 2018

methods have been proposed. While some of these studies deal with categorization of visits based on the user's navigational patters [1, 4, 7, 8], others aim to predict the behavior of users in real time and take actions accordingly to improve the shopping cart abandonment and purchase conversion rates [9–11].

In this paper, we propose a real-time online shopper behavior analysis system. The proposed system consists of two modules which, to the best of our knowledge for the first time, simultaneously predicts visitor's purchasing intention and likelihood to abandon the site. The first module, which assigns a score to the purchasing intention of the visitor in real time during a session, is triggered only if the second module, which predicts the likelihood to abandon the site, produces a greater value than the predetermined threshold. We use an online retailer data and compare the performance of various machine learning algorithms under different conditions.

There are literature studies which aim to categorize the visits based on the user's clickstream data and session information. In one of these studies, Moe [4] aimed to categorize the visits using data from a given online store in the belief that a system could be developed which takes customized actions according to the category of the visit. For this purpose, a set of features were extracted from page-to-page clickstream data of the visits and fed to *k*-means clustering algorithm to categorize the visits according to their purchasing likelihood. The obtained clusters, which are labeled as "Directed Buying," "Hedonic Browsing," "Knowledge Building," "Search/Deliberation," and "Shallow," were determined to have different intentions to purchase when analyzed in terms of the behaviors of the visitors in each cluster. The "Directed Buying" cluster constitutes the group that visited the e-commerce site for direct purchasing purposes, whereas the "Shallow" represents the group of visitors leaving the site after only 2 pageviews. In another study, Mobasher et al. [8] set up two different clustering models based on user transactions and pageviews to derive useful aggregate usage profiles that can be effectively used by recommender systems to take specific actions in real time. The results showed that the profiles extracted from user clickstream data can be helpful in achieving effective personalization at early stages of user's visits in a virtual shopping environment. Such features that were extracted from session information and clickstream data used to group the visits according to the visitor's intention are used to formulate a supervised learning problem in the first module of our system with the aim of estimating the visitor's tendency to finalize the transaction. Thus, we determine the users that visit the e-commerce site with direct purchasing intention and offer content only to those visitors if they are likely to leave the site without finalizing the transaction. We also determine the most discriminative factors in predicting the purchasing intention using filter feature selection techniques.

In a recent study, Suchacka and Chodak [12] aimed to characterize e-customer behaviors based on Web server log data. The dataset was collected from an online bookstore built on an osCommerce platform. A dedicated C++ program was used to extract a set of session features which are session length in terms of the number of Web pages visited in session, session duration in seconds, average time per page in seconds, traffic type representing the page which had referred the user to the bookstore site, three binary variables representing a set of key operations related to the commercial intent, and a set of product categories viewed during the session. They applied association rule mining on this dataset to assess the purchasing probability of the visitors and extract some useful knowledge about the behavior of different customer profiles. In [13], similar to the first module of our system, the prediction of purchasing intention problem was designed as a supervised learning problem and historical data collected from an online bookstore was used to categorize the user sessions as browsing and buyer sessions. Support vector machines (SVMs) with different kernel types were used for classification which is one of the classifiers used in our comparative analysis. In another study, *k*-nearest neighbor (*k*-NN) classifier was used on the same dataset to achieve the same goal [14]. However, considering that *k*-NN is not suitable for real-time prediction since it is a lazy-learning algorithm, it is excluded in the modeling of the purchasing intention task in our study.

The literature studies that aim to predict the behavior of users in real time to be able to take customized actions accordingly mostly use sequential data. Budnikas [10] noted the importance of monitoring real-time behaviors in virtual shopping environment and taking the actions accordingly. Budnikas [10] proposed to classify the visitor behavior patterns with the aim of determining the Web site component that has the highest impact on a fulfillment of business objective. The data set was created using the Google Analytics tracking code [15]. Naïve Bayes and multilayer perceptron classifiers have been used to build a model of consumer on-site behavior to predict whether a Web site guest is eager to finalize a transaction or not.

Yeung [16] also showed that the navigation paths of visitors in the e-commerce site can be used to predict the actions of the visitors. There are many studies that use hidden Markov model (HMM) to determine the frequencies of the paths that are visited consecutively during the session [9, 17, 18]. The most frequently followed navigation paths are used to choose the Web pages the user is likely to visit in the next steps and these pages are recommended to the user to extend the time that he/she will spend in the site.

In another study, considering the loss of throughput in Web servers due to overloading, Poggi et al. [19] proposed a system to assign priorities to sessions according to the revenue that will generate using early clickstream data and session information. The training dataset was created from 7000 transactions, of which half were chosen to be from "buying" class to deal with class imbalance problem. They used Markov chains, logistic linear regression, decision trees, and Naïve Bayes to generate a probability on the users' purchasing intention. Ding et al. [3] used HMM to model the clickstream data of the visitors and showed that predicting the intention of the user in real time and taking customized actions in this context helps to increase the conversion rates and decrease the shopping cart abandonment rates. In our study, we use long short-term memory (LSTM) recurrent neural network (RNN) (LSTM-RNN) instead of HMM to process the clickstream data. This is based on the findings that RNN produces models with higher learning capacity and generalization ability than HMM with the increasing number of samples in the sequence [20]. Although recently a few studies have used RNN to process e-commerce data, these studies focused on session-based recommender systems in which a recommendation is produced after each consecutive click of the user [21]. Unlike these studies, we train LSTM-RNN with sequential clickstream data to predict the probability that the user will leave the site within a certain time.

2 Predicting online purchasing intention

2.1 Dataset description

In our study, the purchasing intention model is designed as a binary classification problem measuring the user's intention to finalize the transaction. Thus, we aim to offer content only to those who intend to purchase and not to offer content to the other users. The numerical and categorical features used in the purchasing intention prediction model are shown in Tables 1 and 2, respectively. The dataset consists of feature vectors belonging to 12,330 sessions. The dataset was formed so that each session would belong to a different user in a 1-year period to avoid any tendency to a specific campaign, special day, user profile, or period. Of the 12,330 sessions in the dataset, 84.5% (10,422) were negative class samples that did not end with shopping, and the rest (1908) were positive class samples ending with shopping.

Table 1 shows the numerical features along with their statistical parameters. Among these features, "Administrative," "Administrative Duration," "Informational," "Informational Duration," "Product Related," and "Product Related Duration" represent the number of different types of pages visited by the visitor in that session and total time spent in each of these page categories. The values of these features are derived from the URL information of the pages visited by the user and updated in real time when a user takes an action, e.g., moving from one page to another. The "Bounce Rate," "Exit Rate," and "Page Value" features shown in Table 1 represent the metrics measured by "Google Analytics" [15] for each page in the e-commerce site. These values can be stored in the application database for all Web pages of the e-commerce site in the developed system and updated automatically at regular intervals. The value of "Bounce Rate" feature for a Web page refers to the percentage of visitors who enter the site from that page and then leave ("bounce") without triggering any other requests to the analytics server during that session. The value of "Exit Rate" feature for a specific Web page is calculated as for all pageviews to the page, the percentage that were the last in the session. The "Page Value" feature represents the average value for a Web page that a user visited before completing an e-commerce transaction. The "Special Day" feature indicates the closeness of the site visiting time to a specific special day (e.g., Mother's Day, Valentine's Day) in which the sessions are more likely to be finalized with transaction. The value of this attribute is determined by considering the dynamics of e-commerce such as the duration between the order date and delivery date. For example, for Valentina's day, this value takes a nonzero value between February 2 and February 12, zero before and after this date unless it is close to another special day, and its maximum value of 1 on February 8.

2.2 Prediction

In the scope of this study, lazy-learning algorithms such as *k*-nearest neighbors are excluded in the modeling of the visitors' purchasing intention, considering the real-time use of the system. Since the system needs to be updated with new examples, multilayer perceptron (MLP) and decision tree algorithms, which have online learning implementations, are selected for comparison. Support vector machine (SVM) classifier is also included in our analysis due to its successful applications in various machine learning applications [22]. The performance of the classification algorithms used in this study is compared using accuracy, F1 Score, and true-positive/true-negative rate evaluation metrics. The experiments were repeated 100 times with randomly chosen training and test instances, and *t* test was applied to test whether the accuracies of the algorithms are significantly different from each other.

Table 1 Numerical features used in the user behavior analysis model

Feature name	Feature description	Min. value	Max. value	SD
Administrative	Number of pages visited by the visitor about account management	0	27	3.32
Administrative duration	Total amount of time (in seconds) spent by the visitor on account management related pages	0	3398	176.70
Informational	Number of pages visited by the visitor about Web site, communication and address information of the shopping site	0	24	1.26
Informational duration	Total amount of time (in seconds) spent by the visitor on informational pages	0	2549	140.64
Product related	Number of pages visited by visitor about product related pages	0	705	44.45
Product related duration	Total amount of time (in seconds) spent by the visitor on product related pages	0	63,973	1912.25
Bounce rate	Average bounce rate value of the pages visited by the visitor	0	0.2	0.04
Exit rate	Average exit rate value of the pages visited by the visitor	0	0.2	0.05
Page value	Average page value of the pages visited by the visitor	0	361	18.55
Special day	Closeness of the site visiting time to a special day	0	1.0	0.19

Table 2 Categorical features used in the user behavior analysis model

Feature name	Feature description	Number of categorical values
OperatingSystems	Operating system of the visitor	8
Browser	Browser of the visitor	13
Region	Geographic region from which the session has been started by the visitor	9
TrafficType	Traffic source by which the visitor has arrived at the Web site (e.g., banner, SMS, direct)	20
VisitorType	Visitor type as "New Visitor," "Returning Visitor," and "Other"	3
Weekend	Boolean value indicating whether the date of the visit is weekend	2
Month	Month value of the visit date	12
Revenue	Class label indicating whether the visit has been finalized with a transaction	2

2.2.1 Multilayer perceptron

MLP is a feedforward artificial neural network model that is made up of multiple layers of nodes in a directed graph, with each layer fully connected to the next one. The elements of the hidden and output layers are called neurons. Each neuron is a processing unit. Our MLP model consists of an input, an output, and a single hidden layer. MLP is capable of modeling complex nonlinear problems with the use of a nonlinear activation function in its hidden layer [23, 24].

In regression, the sum of errors over the whole set of training samples is

$$E(\boldsymbol{W}, \boldsymbol{v}|X) = \sum_{t=1}^{T} (r^t - y^t)^2 \quad (1)$$

where $\boldsymbol{W}$ and $\boldsymbol{v}$ denote the set of first and second layer weights, respectively, T the number of training set samples, r^t the actual value of sample t, and y^t the output of the network, i.e., the predicted value, for sample t. The output of the network is calculated as

$$y^t = \sum_{h=1}^{H} v_h z_h^t + v_0 \quad (2)$$

where v_h denotes the weight between hidden node h and the output, and z_h^t the value of hidden node h for sample t. In a two-class classification problem, the output, y^t is passed through a sigmoid function.

The parameters of the neural network, which are the weights representing the connections between the layers, are learned iteratively during the training process. The weights, $\boldsymbol{W}$ and $\boldsymbol{v}$, are updated according to the rule of a learning algorithm. The traditional learning algorithm used to train the network is backpropagation [25] which updates the weights of a neural network to find a local minimum of the error function given in Eq. 1. The second layer of MLP

is a simple perceptron with hidden units as inputs [26]. Therefore, the least squares rule is used to update the second layer weights:

$$\Delta v_h^t = \eta(r^t - y^t)z_h^t \tag{3}$$

where η is the learning rate used to determine the magnitude of change to be made in the weight. On the other hand, for the second layer weights, the least squares rule cannot be applied directly as the desired outputs for the hidden neurons are not available. Therefore, the error is back-propagated from the output to the inputs using the following chain rule:

$$\frac{\partial E}{\partial w_{hj}} = \frac{\partial E}{\partial y_i}\frac{\partial y_i}{\partial z_h}\frac{\partial z_h}{\partial w_{hj}} \tag{4}$$

and the update rule of the second layer weight for sample t is found as

$$\Delta w_{hj}^t = \eta(r^t - y^t)v_h z_h^t\left(1 - z_h^t\right)x_j^t \tag{5}$$

where w_{hj} denotes the second layer weight between hidden input j and hidden node h, and x_j^t denotes jth feature of input t. The weights are updated in the opposite direction of the partial derivatives until a local minimum is reached. In this study, we use resilient backpropagation with weight backtracking algorithm to calculate the neural network [25]. Resilient backpropagation is known as one of the fastest algorithms used to train a neural network [27–29]. While the traditional backpropagation algorithm has a specific learning rate for all weights, in resilient back-propagation a separate learning rate that can be modified during the training process is used for each weight. This approach addresses the problem of defining an overall learning rate which should be appropriate for the whole training process and the entire network. Besides, in resilient backpropagation, only the sign of the partial derivates is used to modify the weights which ensures that the learning rate has an equal influence over the entire network. Thus, the update rule of the traditional backpropagation given in Eq. 5 is turned into

$$\Delta w_{hj}^t = \eta_{hj}\text{sign}\left((r^t - y^t)v_h z_h^t\left(1 - z_h^t\right)x_j^t\right) \tag{6}$$

where η_{hj} denotes the learning rate between hth hidden node and jth input. The learning rate can dynamically be changed during learning process for faster convergence. This mechanism is called adaptive learning rate. The idea is based on increasing the value of η_{hj} if the corresponding partial derivative keeps its sign, and decreasing it if the partial derivative of the error function changes its sign. Thus, the local minimum missed due to the large value of learning rate is aimed to be reached in the next iteration [27]. The weight backtracking mechanism used in our experiments undoes the last iteration and adds a smaller value to the weight in the next step to avoid jumping over the minimum again in the latter iterations. In our experiments, we present results for various number of hidden neurons in the hidden layer. The number of iterations is dynamically determined using threshold value of 0.2 for the partial derivatives of the error function as stopping criteria.

2.2.2 Support vector machines

Support vector machine (SVM) classifier, whose classification ability has been shown in many literature studies [22], is also included in our analysis. Although SVM does not have a straightforward implementation for online learning, an online passive–aggressive implementation can be used to dynamically update the SVM model with new examples if it achieves significantly higher accuracies than the other classifiers used in this study. SVM is a discriminant-based algorithm which aims to find the optimal separation boundary called hyperplane to discriminate the classes from each other [30]. The closest samples to these hyperplanes are called support vectors, and the discriminant is represented as the weighted sum of this subset of samples which limits the complexity of the problem. The optimization problem to find an optimal separating hyperplane is defined as:

$$\min\frac{1}{2}\left\|w^2\right\| + C\sum_{i=1}^{k}\xi_i \text{ subject to } r^t\left(w^T x^t + w_0\right) \geq 1 - \xi_i \tag{7}$$

where w is a weight vector defining the discriminant, C the regularization parameter, $\xi = (\xi_1, \xi_2, \ldots, \xi_k)$ vector of slack variables, and r^t the actual value of sample t. The slack variables are defined to tolerate the error on training set in order to avoid overfitting and so improve the generalization ability of the model. The regularization (cost) parameter, C, is a hyperparameter of the algorithm which is used to control the complexity of the model that is fitted to the data. Higher values of C decrease the tolerance of the model on training set instances and hence may cause overfitting on the training set.

Although SVM is a linear classifier, it is capable of modeling nonlinear interactions by mapping the original input space into a higher dimensional feature space using a kernel function. Thus, the linear model in the new space corresponds to a nonlinear model in the original space [26]. In this study, linear and radial basis function (RBF) kernels are used. The RBF is defined as

$$K(\boldsymbol{x}^t, \boldsymbol{x}) = \exp\left[-\frac{\|\boldsymbol{x}^t - \boldsymbol{x}\|}{2s^2}\right] \tag{8}$$

where $\boldsymbol{x}^t$ is the center and s defines the radius [26]. As noted in Sect. 3, we repeat train/validation split procedure for 100 times and report the average performance of each classifier on the validation sets. To avoid overfitting and report unbiased results, the values of hyperparameters, C and s, are optimized using grid search on a randomly selected single train/validation partition, and the specified values are used for the rest of the partitions. We used LIBSVM [31] implementation of SVM for experimental analysis.

2.2.3 Decision trees

The other classifiers used to predict the commercial intent of the visitors are the variants of decision tree algorithms. Decision tree is an efficient nonparametric method that can be used for both classification and regression [32]. A decision tree has two main components: internal decision nodes and terminal leaves. Each internal node in the tree implements a test function using one or more features and each branch descending from that node is labeled with the corresponding discrete outcome. During testing, when a new instance is given, the test pointed out by the root node is applied to the instance and according to the output of the decision node the next internal node that will be visited is determined. This process is then repeated for the subtree rooted at the new node until a leaf node is encountered which is the output of the constructed tree for the given test instance. In this paper, we use C4.5 algorithm to generate an individual decision tree for classification [33]. The C4.5 algorithm extended the ID3 tree construction algorithm by allowing numerical attributes, dealing with missing values and performing tree pruning after construction. Random forest is based on constructing a forest, e.g., a set of diverse and accurate classification trees, using bagging resampling technique and combining the predictions of the individual trees using a voting strategy [34]. The steps of the random forest construction algorithm are shown below:

Step 1: Given N instances in the original training set, create a subsample with bagging, i.e., choose N instances at random with replacement from the original data which constitutes the training set.
Step 2: Suppose that each instance is represented with M input variables in the original input space. A number m is specified, which is much less than M, such that at each node, m variables are selected at random out of the M and the best split on these m is used to split the node.
Step 3: According to the predetermined stopping criteria, each tree is grown to the largest extent possible without pruning.
Step 4: Repeat this process until desired number of trees is obtained for the forest.

The random forest algorithm proved itself to be effective for many classification problems such as gene classification [35], remote sensing classification [36], land-cover classification [37], or image classification [38]. Therefore, random forest is determined to be used as another classification algorithm in our purchasing intention prediction module. The hyper-parameters of the algorithm are the size of each bag, number of input variables used to determine the best split in each step which is referred to as m in the above-given algorithm, and number of trees in the forest. In our experiments, m is set to $\lceil \log_2 M \rceil$, the size of each bag is set to N, and the number of trees in the forest to 100.

2.3 Filter-based feature selection

We apply feature selection techniques to improve the classification performance and/or scalability of the system. Thus, we aim to investigate whether better or similar classification performance can be achieved with less number of features. An alternative of feature selection is the use a feature extraction technique such as Principal Component Analysis for dimensionality reduction. However, in this case, the features in the reduced space will be the linear combinations of 17 attributes, which brings the need of tracking all features during the visit and updating the feature vector after a new action is taken by the visitor. Therefore, it has been deemed appropriate to apply feature selection instead of feature extraction within the scope of this study.

For feature ranking, we prefer to apply filter-based feature selection instead of wrapper algorithms that require a learning algorithm to be used and consequently can result in reduced feature sets specific to that classifier [39]. We use correlation, mutual information (MI) and mRMR filters in our experiments.

MI is a measure of mutual dependence of the two variables. While correlation coefficient can only capture linear relations, MI can also capture nonlinear relations. MI is based on Shannon's entropy which is a measure of the uncertainty of a random variable X [40]. Shannon's entropy can be regarded as a measure of how difficult it is to predict that variable. The definition of Shannon's entropy can be written as:

$$H(X) = -E[\log P(X)]| = -\sum_{x} [p(x) \log(p(x))] \tag{9}$$

where $p(x) = P(X = x)$ is the probability distribution function of X. MI is a measure of mutual dependence of the two variables based on the entropy:

$$I(X;Y) = H(X) + H(Y) - H(X,Y) \tag{10}$$

For MI-based feature selection, first we compute the mutual information score between each feature and class label. Then, we rank the features according to their mutual information score in descending order and choose top-n features to be fed to the learning algorithm. We should note that as described in dataset description part, some of the features of the dataset used in this study are continuous. Since MI is defined for discrete variables, the continuous variables should be discretized. In our experiments, we use a binning procedure in which each feature is discretized to 9 discrete levels [40, 41]. For this purpose, first we compute the mean, μ, and standard deviation, σ, of each feature. Then, the four intervals of size σ to the right of $\mu + \sigma/2$ are converted to discrete levels from 1 to 4, and the four intervals of size σ to the left of $\mu - \sigma/2$ are mapped to discrete levels from -1 to -4. While the feature values between $\mu - \sigma/2$ and $\mu + \sigma/2$ are converted to 0, very large positive or negative feature values are truncated and discretized to ± 4 appropriately.

In addition to correlation and MI filters, we also use mRMR algorithm for feature subset selection. In mRMR algorithm [40, 41], the aim is to maximize the relevance between the selected set of features and class variable while avoiding the redundancy among the selected features. Thus, maximum classification accuracy is aimed to be obtained with minimal subset of features. According to mRMR approach, mth feature chosen for inclusion in the set of selected variables must satisfy the below condition:

$$\max_{x_j \in X - S_{m-1}} \left[I(x_j; y) - \frac{1}{m-1} \sum_{x_i \in S_{m-1}} I(x_j; x_i) \right] \tag{11}$$

where X is the whole set of features, y is the class variable, S_{m-1} is the set of selected features containing top-ranked $m-1$ elements, and x_j is the jth candidate feature which has not been selected yet by the algorithm. That is, the relevance term that is computed as the mutual information (MI) between a candidate variable and the class variable is discounted by the redundancy term which is computed as the average MI between the candidate variable and already selected variables. The difference between these two terms can be regarded as the amount of unique information that the candidate variable possesses about the target variable.

3 Predicting likelihood of abandonment

In the proposed system, the algorithm used to decide whether to offer a content during a visit is triggered only if the user is likely to abandon the site without shopping. For this abandonment analysis, a long short-term memory (LSTM) recurrent neural network (RNN) [42–44] model is constructed to foresee the visitors that will leave the site in a certain period, which can be called as prediction horizon. For this purpose, for each pageview action taken by a user during visit, the type of the page and the amount of time spent on the page information is used as the feature vector.

3.1 Dataset description

The dataset used for the recurrent neural network-based abandonment analysis module contains 185,000 Web pages visited in 9800 sessions of 3500 visitors. In this dataset, the "product view," "administrative operation," and "information acquisition operation" types of actions have been extracted from the URL information. Besides, "shopping cart operation," which is supposed to carry very important information for abandonment analysis, has been used as a separate action type. In addition to the page type information that is represented with four binary value obtained using 1-of-C coding, the time spent on the corresponding page is also used as a feature. During a visit, this feature vector is generated after one of these actions is taken by the visitor and fed to LSTM-RNN to process the sequential data.

3.2 RNN-based abandonment analysis module

While a traditional multilayer perceptron has only feedforward connections, units in a recurrent neural network have feedforward connections, self-connections, and connections to units in the previous layers [26]. In addition to the current input instance, recurrent neural networks (RNNs) take what they have perceived previously in time as input which let them to make use of sequential information. More formally, given an input sequence $\boldsymbol{x} = (x_1, x_2, \ldots, x_T)$, the recurrent hidden states at timestamp t, $\boldsymbol{h}_t$, are updated by

$$\boldsymbol{h}_t = f(\boldsymbol{x}_t, \boldsymbol{h}_{t-1}) \tag{12}$$

where $\boldsymbol{h}_{t-1}$ denotes the previous hidden states. The traditional approach used to update the recurrent hidden state given in Eq. (12) is

$$\boldsymbol{h}_t = g(W_{xh}\boldsymbol{x}_t + W_{hh}\boldsymbol{h}_{t-1}) \tag{13}$$

where W_{xh} is the input-to-hidden weight matrix, W_{hh} is the state-to-state recurrent weight matrix, and g is the hidden layer function. A smooth and bounded function such as logistic sigmoid function or a hyperbolic tangent function is used as the hidden layer function [26, 45].

The probability of an input sequence, $\boldsymbol{x} = (x_1, x_2, \ldots, x_T)$, can be factorized into

$$p(x_1, x_2, \ldots, x_T) = p(x_1)p(x_2|x_1)\ldots p(x_T|x_1, x_2, \ldots, x_{T-1}). \tag{14}$$

Then, given the current symbol x_t and hidden states $\boldsymbol{h}_t$, a

generative RNN predicts the probability of the next symbol x_{t+1} with

$$p(x_{t+1}|x_1, x_2, \ldots, x_t) = g(\boldsymbol{h}_t) \tag{15}$$

where $\boldsymbol{h}_t$ is computed from Eq. (12).

RNNs have successfully been applied to recognize patterns in sequential or time series data such as handwriting, genomes, speech, image, or stock market [46]. In addition to the applications in which RNNs are used to process time sequences, model-based RNNs that are training free have been successfully applied to different problems such as the control of redundant manipulators [47, 48]. RNNs have also successful model-based applications Although RNNs, in theory, are designed to handle long-term dependencies, it has been shown that due to vanishing gradient problem [49, 50], which occurs when backpropagating errors across many time steps, they have difficulties in learning dependencies between steps that are far apart [46]. To overcome this problem, Hochreiter and Schmidhuber [44] proposed an architecture called long short-term memory (LSTM), in which each traditional node in the hidden layer is replaced by an LSTM unit which consists of a memory cell and three types of gates. The gates are included to protect and control the state of the cell.

The content of the memory cell is updated with

$$c_t^j = f_t^j c_{t-1}^j + i_t^j \tilde{c}_t^j \tag{16}$$

where c_t^j is the memory cell of the jth LSTM unit at timestep t, and i_t^j and f_t^j are the input and forget gates, respectively. While the input gate controls how much latest content should be memorized, the forget gate modulates the extent to which the existing memory is forgotten. The hidden state of jth LSTM unit at timestep t is computed as

$$h_t^j = o_t^j \tan\left(c_t^j\right) \tag{17}$$

where o_t^j is an output gate that controls the amount of memory content exposure. The input, forget, and output gates are computed based on the previous hidden states and the current input:

$$\begin{aligned} \boldsymbol{i}_t &= \sigma(W_{xi}\boldsymbol{x}_t + W_{hi}\boldsymbol{h}_{t-1}) \\ \boldsymbol{f}_t &= \sigma\left(W_{xf}\boldsymbol{x}_t + W_{hf}\boldsymbol{h}_{t-1}\right) \\ \boldsymbol{o}_t &= \sigma(W_{xo}\boldsymbol{x}_t + W_{ho}\boldsymbol{h}_{t-1}) \end{aligned} \tag{18}$$

where σ is the sigmoid function, W terms denote the weight matrices, $\boldsymbol{x}_t$ is the input vector and $\boldsymbol{h}_{t-1}$ is the previous hidden state.

The aim of abandonment analysis module is to foresee that the user is about to abandon the site using the navigational clickstream data before he/she steps into exit action. For this purpose, on the basis of the findings in the related literature [4, 17] that the visitors' navigation path during a visit in an e-commerce site can be used to predict the actions that will be taken by the user, we process the sequential pageview patterns as a time series data and aim to predict whether the visitor will leave the site within a certain period of time. In the literature, most of the related studies address this issue using hidden Markov model (HMM). Unlike these studies, we construct a LSTM-RNN model to process this sequential data considering the average number of Web pages visited in each session since with the increasing number of samples in the sequence, RNN produces models with higher learning capacity and generalization ability than HMM [18]. We should note that HMM has better scalability in the training phase than RNN. However, for training we improve the scalability of RNN using its GPU implementation [51]. During real-time prediction, using LSTM-RNN is feasible since only test is performed after each action is taken by the user and the system is planned to be dynamically updated with new examples outside of active hours using online learning implementation. In order to show the effectiveness of LSTM-RNN, we compare its overall accuracy with three other neural network models which are MLP, extreme learning machine, and radial basis functions. In these neural network models, we use time-delay approach to model the dependencies between the last three actions of the visitor.

4 Proposed system

The activity diagram of the proposed system is shown in Fig. 1. Starting from the landing page, the values of the features determined to be used in purchasing intention and abandonment modules are kept track and updated after each pageview. The updated feature vector is fed to LSTM-RNN test script which generates a sigmoid output showing the probability estimate of visitor's intention to leave the site without finalizing the transaction. If the output of this module is greater than the predetermined threshold, the sigmoid output of the MLP script is checked. If the output of the MLP is greater than the predetermined threshold (default value is 0.5), a content is offered to the visitor to enable him/her to continue browsing and finalize the transaction. In summary, the steps of the algorithm are shown in Algorithm 1.

In the e-commerce Web site abandonment analysis, it has been shown in the literature that the minimum number of pages that should be displayed by the visitor to be able to design an effective learning problem is three [3]. Based on this finding, the step size of the RNN model has been set to three. Our LSTM-RNN network consists of a single hidden layer containing 30 neurons. When the system is designed as a supervised learning problem, different approaches can be considered for the prediction of

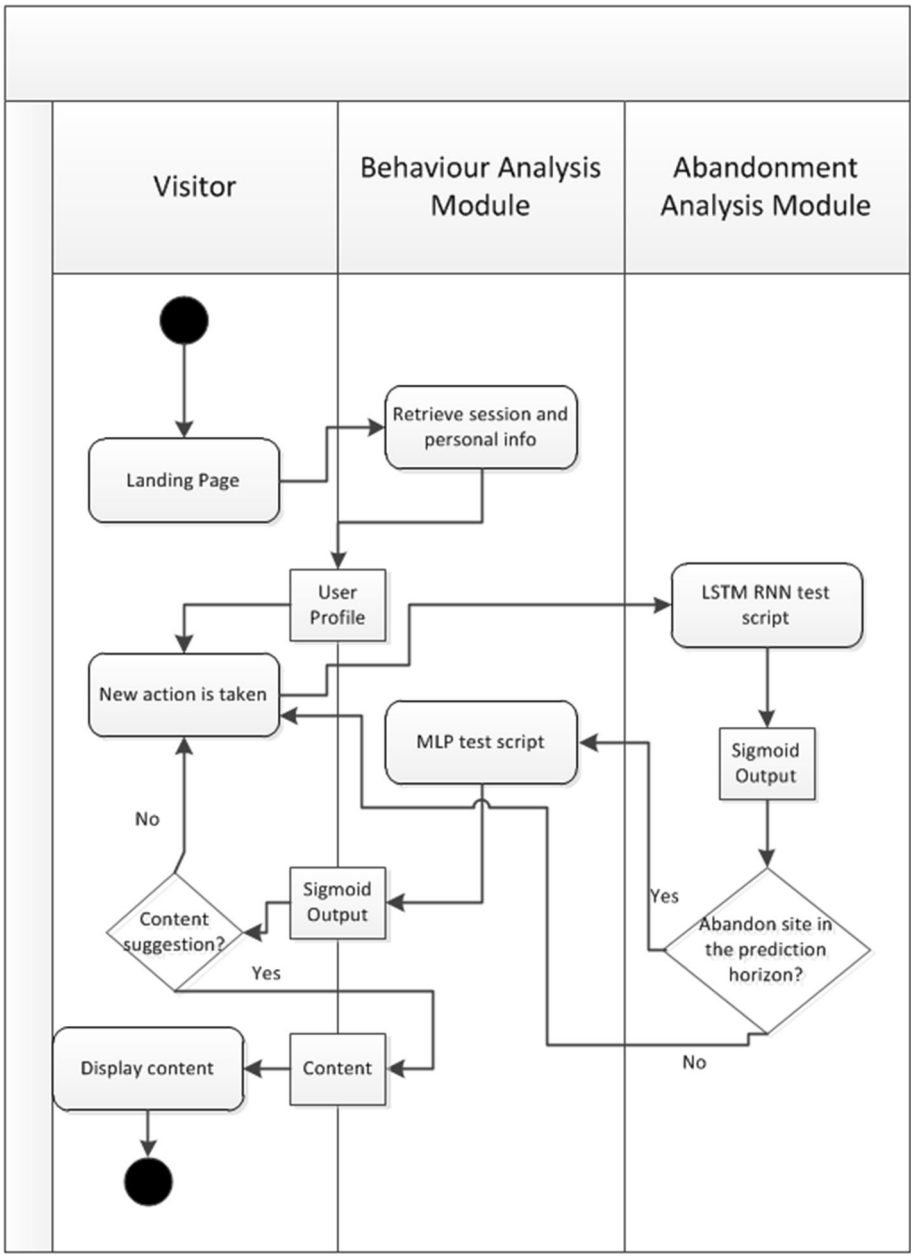

Fig. 1 Activity diagram of the proposed system

abandonment action. The first of these approaches is whether the visitor will leave the site in a "prediction horizon" determined in seconds. The second approach is that whether the visitor will abandon the site in a "prediction horizon" in terms of the page views. In the third approach, the problem can be designed as a supervised regression problem in which the estimation is performed based on the number of seconds that the user will stay in site. In this study, we prefer to use the second approach based on the analysis of the distribution of the total number of seconds that the visitors stay in site in our dataset. Thus, the prediction horizon is determined as the number of pageview actions that will be taken by the visitor before leaving the Web site.

4.1 Predicting purchasing intention

As preprocessing steps, the categorical variables are mapped to 1-of-C coding and the numerical features are standardized to zero mean and unit variance. Then, the dataset is fed to decision tree, support vector machines, and multilayer perceptron classifiers using 70% of dataset for training and the rest for validation. For statistical significance, this procedure is repeated 100 times with random training/validation partitions, and t test is applied to test

whether the performance of the algorithms is statistically different from each other. We present the average accuracy, true-positive rate, true-negative rate, and F1 Score for each classifier.

Algorithm 1: User Behaviour Analysis

Given initial clickstream data X_1; session information data X_2; α_1 abandonment threshold; α_2 purchasing intention threshold

repeat (for each action taken by visitor)
  update X_1
  Standardize each column of X_1
  $s_1 = \text{lstm\_rnn}(X_1)$ //abandonment analysis module to estimate the abandonment likelihood
  if $s_1 > \alpha_1$ //visitor is likely to leave the website in the prediction horizon
    $s_2 = \text{mlp}([X_1 X_2])$ //behaviour analysis module to estimate the purchasing intention
    if $s_2 > \alpha_2$ //visitor has purchasing intention
      display content to visitor
      terminate
until session is terminated by the visitor

4.1.1 Results on class imbalanced dataset

Tables 3, 4, and 5 show the results obtained with decision tree, MLP, and SVM algorithms on the test set, respectively. The results show that C4.5 implementation of the decision tree algorithm gives the highest accuracy rate on test set. However, a class imbalance problem arises [52] since the number of negative class instances in the data set is much higher than that of the positive class instances, and the imbalanced success rates on positive (TPR) and negative (TNR) samples show that the classifiers tend to label the test samples as the majority class. This class imbalance problem is a natural situation for the analyzed problem since most of the e-commerce visits do not end with shopping [3]. Therefore, alternative metrics that take the class imbalanced into account such as F1 Score should be used to evaluate the performance of the classifiers. In this study, the results are reported together with the general accuracy rate as well as the individual class accuracies and F1 Score, as shown in the tables. The highest F1 Score of 0.58 was obtained with multilayer perceptron (MLP) having 70 neurons in its input layer obtained by mapping the categorical variables to 1-of-C coding and 20 neurons in its hidden layer. On the other hand, the lowest F1 Score (0.52) was obtained with linear kernel SVM. As it is seen in Table 5, although linear kernel SVM gives significantly higher accuracy than RBF kernel SVM, RBF kernel SVM produces more balanced classification rates on positive and negative samples.

4.1.2 Results obtained with oversampling

The results presented in Sect. 4.1.1 show that the classifiers tend to minimize their errors on majority class samples, which leads to an imbalance between the accuracy rates of the positive and negative classes. However, in a real-time user behavior analysis model, correctly identifying directed buying visits, which are represented with positive class in our dataset, is as important as identifying negative class samples. Therefore, a balanced classifier is needed to increase the conversion rates in an e-commerce Web site. To deal with class imbalance problem, we use oversampling method, in which a uniform distribution over the classes is aimed to be achieved by adding more of the minority (positive class in our dataset) class instances. Since this dataset is created by selecting multiple instances of the minority class more than once, first oversampling the dataset and then dividing it into training and test sets may lead to biased results due to the possibility that the same minority class instance may be used both for training and test. For this reason, in our study, 30% of the data set consisting of 12,330 samples is first left out for testing and the oversampling method is applied to the remaining 70% of the samples.

Table 3 Results obtained with decision tree-based classifiers on test set

Classifier	Accuracy (%)	True-positive rate (TPR)	True-negative rate (TNR)	F1 Score
C4.5	88.92	0.57	0.96	0.57
Random forest	89.51	0.57	0.96	0.58

$^{**}p < 0.01$; $^{*}p < 0.05$

Table 4 Results obtained with multilayer perceptron on test set

# of neurons in hidden layer	Accuracy (%)	True-positive rate (TPR)	True-negative rate (TNR)	F1 Score
10	87.45	0.53	0.94	0.56
20	87.92	0.56	0.96	0.58
40	87.02	0.54	0.93	0.56

$^{**}p < 0.01$; $^{*}p < 0.05$

Table 5 Results obtained with support vector machines on test set

Kernel	Accuracy (%)	True-positive rate (TPR)	True-negative rate (TNR)	F1 Score
Linear	88.25*	0.42	0.97	0.52
RBF	86.14	0.46	0.92	0.53

$^{**}p < 0.01$; $^{*}p < 0.05$

The results obtained on the balanced dataset are shown in Tables 6, 7, and 8. Since the number of samples belonging to positive and negative classes is equalized with oversampling, both accuracy and F1 Score metrics can be used to evaluate the results. As it is seen, the highest accuracy of 87.24% and F1 Score of 0.86 is obtained with MLP having 10 neurons in its hidden layer. The summary of the results obtained with the best settings of all algorithms is shown in Table 9. It is seen that SVM with RBF kernel gives significantly higher accuracies than C4.5 implementation of decision tree.

4.1.3 Feature selection

The MLP algorithm, which achieved the highest accuracy and F1 Score, has been chosen to identify the directed buying visits. In this section, we apply feature selection to further improve the classification performance of MLP classifier. Besides, considering the real-time usage of the proposed system, achieving better or similar classification performance with less number of features will improve the scalability of the system since less number of features will be kept track during the session.

Table 10 shows the feature rankings obtained with the filters used in this study. The results showed that the "Page Value" feature of Google Analytics tracking tool is selected in the first place by all filters and carries discriminative information about the intent of the visitor. Considering that the "Page Value" [15] feature represents the average value for a page that a user visited before completing an e-commerce transaction, it can be seen as a natural measure of visitor's transaction finalization intention. In our system, this feature is represented as the average of the "Page Value" values of pages visited by the visitor during the session and is updated when the visitor moves to another page. As seen in Table 10, the other two Google Analytics features, "Exit Rate" and "Bounce Rate," are also highly correlated with the class variable and take place near the top in correlation and mutual information filter rankings. However, since "Bounce Rate" is also highly correlated with "Exit Rate" and so contains redundant information, mRMR, which suggests incrementally selecting the maximally relevant variables while avoiding the redundant ones, chooses it in the 15th order. Similarly, although the "Product Related" and "Product Related Duration" attributes are closely related to the class variable, they have been ranked in the last orders by mRMR because of their high correlation with "Page Value" feature which has already been chosen by the algorithm. It is seen that correlation and mutual information filters give similar rankings since both algorithms ignore the relations among the selected variables, whereas the mRMR method gives a quite different ranking compared to these methods.

The top-10 features selected by the filters are incrementally fed to the MLP algorithm using the oversampled dataset. The highest accuracy obtained by each method and the number of input variables in the corresponding MLP model are shown in Table 11. The highest accuracy (87.94%) and F1 Score (0.87) are obtained using the feature subset containing the top 6 features of the mRMR ranking. These values are statistically different from the highest accuracy and F1 Score achieved by the other two methods (p value < 0.05). It is also seen that the correlation and mutual information filters use more features than mRMR algorithm in their best models. Since mRMR filter performs significantly higher accuracy with less number of features than correlation and mutual information, MLP with top-6 features selected by mRMR is determined as the final model considering its better performance as well as scalability of the real-time storage and update of the feature vector periodically during the session.

Table 6 Test set results obtained with decision tree-based classifiers using oversampled dataset

Classifier	Accuracy (%)	True-positive rate (TPR)	True-negative rate (TNR)	F1 Score
C4.5	82.34	0.79	0.85	0.82
Random forest	82.29	0.74	0.90	0.81

$^{**}p < 0.01$; $^{*}p < 0.05$

Table 7 Test set results obtained with multilayer perceptron using oversampled dataset

# of neurons in hidden layer	Accuracy (%)	True-positive rate (TPR)	True-negative rate (TNR)	F1 Score
10	87.24*	0.84	0.92*	0.86*
20	83.84	0.84	0.83	0.83
40	82.15	0.82	0.83	0.82

$^{**}p < 0.01$; $^{*}p < 0.05$

Table 8 Test set results obtained with support vector machines using oversampled dataset

Kernel	Accuracy (%)	True-positive rate (TPR)	True-negative rate (TNR)	F1 Score
Linear	84.26	0.75	0.93	0.82
RBF	84.88	0.75	0.94	0.82

$^{**}p < 0.01$; $^{*}p < 0.05$

Table 9 Summary of best results obtained with the classifiers used in this study

	C4.5 (1)	MLP (2)	RBF SVM (3)	Statistical significance 1–2	Statistical significance 1–3	Statistical significance 2–3
Accuracy (%)	82.34	87.24	84.88	**	*	*
TPR	0.79	0.84	0.75	**	*	**
TNR	0.85	0.92	0.94	**	*	*
F1 Score	0.82	0.86	0.82	**	N.S.	**

N.S. not significant

$^{**}p < 0.01$; $^{*}p < 0.05$

4.2 Predicting likelihood of abandonment

As stated in Sect. 4, the minimum number of pages that should be displayed by the visitor to be able to design an effective learning problem is determined as three in the literature. Based on this finding, the step size of the LSTM-RNN model has been set to three. In addition to LSTM-RNN, we present the results obtained with three other neural network models which are MLP, extreme learning machine (ELM), and radial basis functions (RBF). These models are constructed using time-delay approach by augmenting the current input, which is the type of the Web page visited by the visitor and time spent on that page, with time-delayed copies of previous inputs, which are the types of the last two Web pages and the time spent on each of these Web pages. The number of hidden neurons in the hidden layer of MLP, ELM, and RBF models are 10, 75, and, 8, respectively.

Figure 2 shows the overall accuracy and true-positive rates obtained on the test set with respect to the prediction horizon which is determined as the number of actions that will be taken by the visitor before leaving the Web site. We should note that the test set is constituted so that it contains equal number of positive and negative samples for all prediction horizon values. As seen in Fig. 2, LSTM-RNN model correctly predicts that the user will leave the Web site after a single action with 74.3% accuracy. It is also observed that for all prediction horizon settings, LSTM-RNN produces significantly (p value < 0.05) higher accuracies than the other neural network types. Figure 2 shows that as the prediction horizon increases, the success rate of the abandonment prediction model decreases since the time

Table 10 Feature rankings obtained with filter feature selection methods

Ranking	Correlation	Mutual information	mRMR
1	Page value	Page value	Page value
2	Exit rate	Exit rate	Month
3	Product related	Product related duration	Exit rate
4	Product related duration	Bounce Rate	Weekend
5	Bounce rate	Product related	Informational duration
6	Administrative	Traffic type	Region
7	Visitor type	Administrative	Operating system
8	Informational	Month	Administrative duration
9	Administrative duration	Administrative duration	Visitor type
10	Special day	Informational	Product related duration
11	Month	Informational duration	Special day
12	Traffic type	Visitor type	Informational
13	Informational duration	Special day	Traffic type
14	Operating system	Operating system	Administrative
15	Weekend	Browser	Bounce rate
16	Region	Weekend	Browser
17	Browser	Region	Product related

Table 11 Best results obtained by feeding top-ranked features as input to MLP

Filter	Number of features	Accuracy (%)	True-positive rate (TPR)	True-negative rate (TNR)	F1 Score
Correlation	9	84.32	0.84	0.85	0.84
Mutual information	10	84.11	0.84	0.85	0.84
mRMR	6	87.94*	0.84	0.92**	0.87*

$^{**}p < 0.01$; $^{*}p < 0.05$

and the number of actions to the user's final exit decision also increases.

The MLP model, which is used to determine whether the user should be offered content, is triggered when the output of the RNN-based abandonment analysis model (referred to as s_1 in Algorithm 1) exceeds the threshold value (referred to as α_1 in Algorithm 1) set after a minimum of three pages have been visited. Figure 2 (right) shows the true-positive rates obtained with LSTM-RNN for various threshold values. It is seen that the maximum true-positive rate of 0.94 is achieved when the prediction horizon and threshold values are set to 2 and 0.90, respectively. As the threshold value increases, the tendency of the system to suggest campaigns decreases since the system rarely produces a positive prediction. The determined threshold value is compared to the sigmoid output of the RNN network. For example, for a threshold value of 0.7, if the RNN sigmoid output yields a value of 0.7 or higher, abandonment analysis model produces a positive prediction, and the content suggestion decision is given based on the output of the purchasing intention module.

5 Conclusion

In this paper, we construct a real-time user behavior analysis system for virtual shopping environment which consists of two modules. We use an online retailer data to perform the experiments. In the first module, to predict the purchasing intention of the visitor we use aggregated pageview data kept track during the visit along with some session and user information as input to machine learning algorithms. We apply oversampling and feature selection preprocessing techniques to improve the success rates and scalability of the algorithms. The best results are achieved with a multilayer perceptron network calculated using resilient backpropagation with weight backtracking. In the second module, using only sequential clickstream data, we train a long short-term memory-based recurrent neural network (LSTM-RNN) that generates a sigmoid output showing the probability estimate of visitor's intention to leave the site without finalizing the transaction in a prediction horizon. The modules simultaneously predict visitor's purchasing intention and likelihood to leave the site.

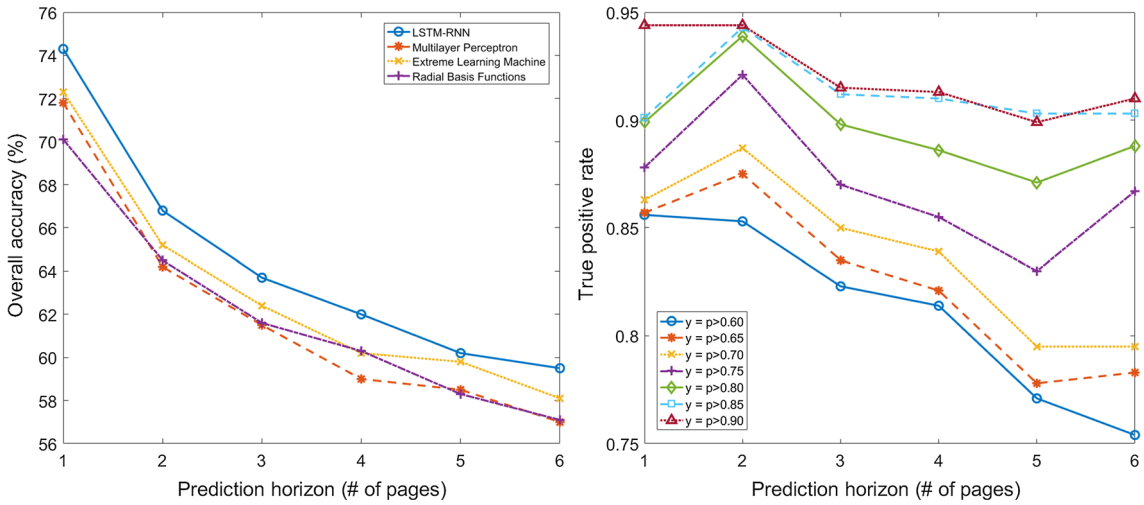

Fig. 2 Results of abandonment prediction module (left) overall accuracy of LSTM-RNN, multilayer perceptron, extreme learning machine, and radial basis function networks in predicting whether a visitor will leave the site in the prediction horizon which is determined as the number of actions that will be taken by the visitor before leaving the Web site (right) True-positive rates produced by LSTM-RNN network with respect to various values of prediction horizon and threshold values

The first module is triggered only if the second module produces a greater value than the predetermined threshold, and accordingly the system decides whether to offer a content during a visit. Thus, we aim to determine the visitors which have purchasing intention and offer content only to those visitors if they are likely to leave the site in the prediction horizon.

Our findings support the argument that the features extracted from clickstream data during the visit convey important information for online purchasing intention prediction. The features that represent aggregated statistics of the clickstream data obtained during the visit are ranked near the top by the filter feature ranking algorithms. However, these metrics are also highly correlated with each other. On the other hand, although the session information-based features are less correlated with purchasing intention of the visitor, they contain unique information different from clickstream-based features. Therefore, we apply a feature ranking method called minimum redundancy–maximum relevance which takes such redundancies between the features into account. The findings show that choosing a minimal subset of combination of clickstream data aggregated statistics and session information such as the date and geographic region results in a more accurate and scalable system. Considering the real-time usage of the proposed system, achieving better or similar classification performance with minimal subset of features can be seen as an important factor since less number of features will be kept track during the session.

In the second module of the proposed system, an LSTM-RNN is trained using only sequential clickstream data to predict the probability that the user will leave the site in the determined prediction horizon. The prediction horizon is determined as the number of pageview actions that will be taken by the visitor before leaving the Web site. An important observation is that as the prediction horizon increases, the time and the number of actions to the user's final exit decision increase, so the estimation becomes more difficult and the success rate is falling. We should also note that the maximum accuracy on the prediction of positive examples is achieved when the prediction horizon is set to 2 pageviews. As a future direction, an item or user-based recommender system may be integrated to the system to further increase the conversion rates by offering user-specific contents to the users who visit the site with purchasing intention and is likely to leave the site in a prediction horizon.

Acknowledgements We would like to thank Gözalan Group (http://www.gozalangroup.com.tr/) for sharing columbia.com.tr data and Inveon analytics team for their assistance throughout this process.

Funding This work was supported by TUBITAK-TEYDEB program under the Project No. 3150945.

Compliance with ethical standards

Conflict of interest The authors declare that they have no conflict of interest.

References

1. Carmona CJ, Ramírez-Gallego S, Torres F, Bernal E, del Jesús MJ, García S (2012) Web usage mining to improve the design of an e-commerce website: OrOliveSur. com. Expert Syst Appl 39(12):11243–11249

2. Rajamma RK, Paswan AK, Hossain MM (2009) Why do shoppers abandon shopping cart? Perceived waiting time, risk, and transaction inconvenience. J Prod Brand Manag 18(3):188–197
3. Ding AW, Li S, Chatterjee P (2015) Learning user real-time intent for optimal dynamic web page transformation. Inf Syst Res 26(2):339–359
4. Moe WW (2003) Buying, searching, or browsing: differentiating between online shoppers using in-store navigational clickstream. J Consum Psychol 13(1–2):29–39
5. Albert TC, Goes PB, Gupta A (2004) A model for design and management of content and interactivity of customer-centric web sites. MIS Q 28(2):161–182
6. Cho CH, Kang J, Cheon HJ (2006) Online shopping hesitation. CyberPsychol Behav 9(3):261–274
7. Keng Kau A, Tang YE, Ghose S (2003) Typology of online shoppers. J Consum Mark 20(2):139–156
8. Mobasher B, Dai H, Luo T, Nakagawa M (2002) Discovery and evaluation of aggregate usage profiles for web personalization. Data Min Knowl Discov 6(1):61–82
9. Awad MA, Khalil I (2012) Prediction of user's web-browsing behavior: application of markov model. IEEE Trans Syst Man Cybern B Cybern 42(4):1131–1142
10. Budnikas G (2015) Computerised recommendations on e-transaction finalisation by means of machine learning. Stat Transit New Ser 16(2):309–322
11. Fernandes RF, Teixeira CM (2015) Using clickstream data to analyze online purchase intentions. Master's thesis, University of Porto
12. Suchacka G, Chodak G (2017) Using association rules to assess purchase probability in online stores. IseB 15(3):751–780
13. Suchacka G, Skolimowska-Kulig M, Potempa A (2015) Classification of e-customer sessions based on support vector machine. ECMS 15:594–600
14. Suchacka G, Skolimowska-Kulig M, Potempa A (2015) A k-nearest neighbors method for classifying user sessions in e-commerce scenario. J Telecommun Inf Technol 3:64
15. Clifton B (2012) Advanced web metrics with Google Analytics. Wiley, New York
16. Yeung WL (2016) A review of data mining techniques for research in online shopping behaviour through frequent navigation paths. HKIBS working paper series 075-1516. Retrieved from Lingnan University website: http://commons.ln.edu.hk/hkibswp/76. Accessed 2 Feb 2018
17. Shi Y, Wen Y, Fan Z, Miao Y (2013) Predicting the next scenic spot a user will browse on a tourism website based on Markov prediction model. In 2013 IEEE 25th international conference on tools with artificial intelligence (ICTAI), pp 195–200
18. Narvekar M, Banu SS (2015) Predicting user's web navigation behavior using hybrid approach. Procedia Comput Sci 45:3–12
19. Poggi N, Moreno T, Berral JL, Gavaldà R, Torres J (2007) Web customer modeling for automated session prioritization on high traffic sites. In: International conference on user modeling. Springer, Berlin, pp 450–454
20. Panzner M, Cimiano P (2016) Comparing hidden Markov models and long short term memory neural networks for learning action representations. In: International workshop on machine learning, optimization and big data. Springer, Cham, pp 94–105
21. Hidasi B, Karatzoglou A, Baltrunas L, Tikk D (2015) Session-based recommendations with recurrent neural networks. arXiv preprint arXiv:1511.06939
22. Salcedo-Sanz S, Rojo-Álvarez JL, Martínez-Ramón M, Camps-Valls G (2014) Support vector machines in engineering: an overview. Wiley Interdiscip Rev Data Min Knowl Discov 4(3):234–267
23. Hornik K, Stinchcombe M, White H (1989) Multilayer feedforward networks are universal approximators. Neural Netw 2(5):359–366
24. Warner B, Misra M (1996) Understanding neural networks as statistical tools. Am Stat 50(4):284–293
25. Riedmiller M, Braun H (1993) A direct adaptive method for faster backpropagation learning: the RPROP algorithm. In: IEEE international conference on neural networks, 1993. IEEE, pp 586–591
26. Alpaydin E (2014) Introduction to machine learning. MIT Press, Cambridge
27. Günther F, Fritsch S (2010) neuralnet: training of neural networks. R J 2(1):30–38
28. Schiffmann W, Joost M, Werner R (1994) Optimization of the backpropagation algorithm for training multilayer perceptrons. University of Koblenz, Koblenz
29. Azar AT (2013) Fast neural network learning algorithms for medical applications. Neural Comput Appl 23(3–4):1019–1034
30. Vapnik V (2013) The nature of statistical learning theory. Springer, Berlin
31. Hsu CW, Lin CJ (2002) A comparison of methods for multiclass support vector machines. IEEE Trans Neural Netw 13(2):415–425
32. Tan PN (2006) Introduction to data mining. Pearson Education, New Delhi
33. Quinlan JR (1993) C4.5: programming for machine learning. San Mateo, Morgan Kauffmann, p 38
34. Breiman L (2001) Random forests. Mach Learn 45(1):5–32
35. Díaz-Uriarte R, De Andres SA (2006) Gene selection and classification of microarray data using random forest. BMC Bioinform 7(1):3
36. Pal M (2005) Random forest classifier for remote sensing classification. Int J Remote Sens 26(1):217–222
37. Rodriguez-Galiano VF, Ghimire B, Rogan J, Chica-Olmo M, Rigol-Sanchez JP (2012) An assessment of the effectiveness of a random forest classifier for land-cover classification. ISPRS J Photogramm Remote Sens 67:93–104
38. Bosch A, Zisserman A, Munoz X (2007) Image classification using random forests and ferns. In: IEEE 11th international conference on computer vision, 2007. ICCV 2007. IEEE, pp 1–8
39. Chandrashekar G, Sahin F (2014) A survey on feature selection methods. Comput Electr Eng 40(1):16–28
40. Peng H, Long F, Ding C (2005) Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy. IEEE Trans Pattern Anal Mach Intell 27(8):1226–1238
41. Sakar CO, Kursun O, Gurgen F (2012) A feature selection method based on kernel canonical correlation analysis and the minimum redundancy-maximum relevance filter method. Expert Syst Appl 39(3):3432–3437
42. Jain LC, Seera M, Lim CP, Balasubramaniam P (2014) A review of online learning in supervised neural networks. Neural Comput Appl 25(3–4):491–509
43. Williams RJ, Zipser D (1989) A learning algorithm for continually running fully recurrent neural networks. Neural Comput 1(2):270–280
44. Hochreiter S, Schmidhuber J (1997) Long short-term memory. Neural Comput 9(8):1735–1780
45. Graves A, Mohamed AR, Hinton G (2013) Speech recognition with deep recurrent neural networks. In: 2013 IEEE international conference on acoustics, speech and signal processing (ICASSP). IEEE, pp 6645–6649
46. Lipton ZC, Berkowitz J, Elkan C (2015) A critical review of recurrent neural networks for sequence learning. arXiv preprint arXiv:1506.00019

47. Li S, Zhang Y, Jin L (2017) Kinematic control of redundant manipulators using neural networks. IEEE Trans Neural Netw Learn Syst 28(10):2243–2254
48. Li S, He J, Li Y, Rafique MU (2017) Distributed recurrent neural networks for cooperative control of manipulators: a game-theoretic perspective. IEEE Trans Neural Netw Learn Syst 28(2):415–426
49. Bengio Y, Simard P, Frasconi P (1994) Learning long-term dependencies with gradient descent is difficult. IEEE Trans Neural Netw 5(2):157–166
50. Hochreiter S, Bengio Y, Frasconi P, Schmidhuber J (2001) Gradient flow in recurrent nets: the difficulty of learning long-term dependencies. In: Kremer SC, Kolen JF (eds) A field guide to dynamical recurrent neural networks. IEEE Press
51. Abadi M, Barham P, Chen J, Chen Z, Davis A, Dean J, Devin M, Ghemawat S, Irving G, Isard M, Kudlur M (2016) TensorFlow: a system for large-scale machine learning. In: Proceedings of the 12th USENIX symposium on operating systems design and implementation (OSDI), Savannah, USA
52. Tian J, Gu H, Liu W (2011) Imbalanced classification using support vector machine ensemble. Neural Comput Appl 20(2):203–209