Context pre-modeling: an empirical analysis for classification based user-centric context-aware predictive modeling

Iqbal H. Sarker[1,2]*i , Hamed Alqahtani[3], Fawaz Alsolami[4], Asif Irshad Khan[4], Yoosef B. Abushark[4] and Mohammad Khubeb Siddiqui[5]

*Correspondence:
msarker@swin.edu.au
[1] Swinburne University
of Technology, Melbourne,
VIC 3122, Australia
Full list of author information
is available at the end of the
article

Abstract

Nowadays, machine learning classification techniques have been successfully used while building data-driven intelligent predictive systems in various application areas including smartphone apps. For an effective context-aware system, *context pre-modeling* is considered as a key issue and task, as the representation of contextual data directly influences the predictive models. This paper mainly explores the role of major context pre-modeling tasks, such as *context vectorization* by defining a good numerical measure through transformation and normalization, *context generation and extraction* by creating new brand principal components, *context selection* by taking into account a subset of original contexts according to their correlations, and eventually *context evaluation*, to build effective context-aware predictive models utilizing multi-dimensional contextual data. For creating models, various popular *machine learning classification* techniques such as decision tree, random forest, k-nearest neighbor, support vector machines, naive Bayes classifier, and deep learning by constructing a neural network of multiple hidden layers, are used in our study. Based on the context pre-modeling tasks and classification methods, we experimentally analyze user-centric smartphone usage behavioral activities utilizing their contextual datasets. The effectiveness of these machine learning context-aware models is examined by considering prediction accuracy, in terms of precision, recall, f-score, and ROC values, and has been made an empirical discussion in various dimensions within the scope of our study.

Keywords: Machine learning, Feature engineering, User behavior modeling, Context-aware computing, Classification, Smartphone data analytics, Predictive analytics, Intelligent systems, IoT analytics and services

Introduction

In the context of today's computing, context-awareness becomes one of the most popular terms, because of the vast usage of Internet of Things (IoT), and lots of applications related to IoT. In particular, with the recent advanced features in the most popular IoT device, i.e., smartphones, context-awareness has become more effective in our daily activities. In the real world, users' interest in "Mobile Phones" is more and more than other platforms like "Desktop Computer", "Laptop Computer" or "Tablet Computer"

Sarker *et al. J Big Data*   (2020) 7:51

Page 2 of 23

over time [1]. Although voice communication is the main activity with one's mobile phone, people use smartphones for various daily activities with apps like social networking systems, online shopping, recommendation systems, instant messaging, tourist guides, location tracking, or medical appointments etc [2]. Individual' behavior with these apps are not static, may vary from user-to-user according to their current needs. Thus, user-centric *context-aware predictive model* with these apps is needed that considers user's current needs in different contexts such as a temporal context that represents time-of-the-days or days-of-the-week, one's working status in workday or holiday, spatial context or user current location, user emotional state, Internet connectivity or Wifi status, or device configuration or relevant status, etc. These contexts may have different types of values depending on individuals' interests and their *behavioral patterns* with the surrounding environment and contexts. Therefore, *context pre-modeling tasks* based on this contextual information is considered as a key issue and task, to build an effective machine learning-based *context-aware model* that assist the users in various day-to-day situations in their daily life activities.

In this paper, we define *context pre-modeling* as the representation of contextual data to build effective context-aware predictive models based on machine learning classification techniques. A "context" in a context-aware modeling is considered as a contextual variable [1], and "context pre-modeling" is the general term for the process of creating and manipulating contextual variables, so that machine learning-based context-aware models can be built. This paper mainly explores the role of major context pre-modeling tasks, such as context vectorization, context generation and extraction, context selection, and eventually context evaluation that are involved in a data-driven context-aware predictive model. In the process of *context vectorization*, the contexts are defined as good numerical measures through transformation and normalization. In *context generation and extraction*, new brand components that capture most of the useful information are created and used in the machine learning classification based modeling. In the process of *context selection* a subset of most relevant contexts is selected, where the less significant, irrelevant, or redundant contexts are eliminated from the dataset. Thus, the key difference between context generation and selection is that context generation creates brand new ones, while context selection keeps a subset of the original contexts according to their relevance and influence with the target behavioral activities of the users. Both approaches could be useful for handling high dimensions of contexts in terms of reducing context-dimensions, model complexity while building effective context-aware models and systems, as well as increasing prediction accuracy with unseen test cases. Finally, in the process of *context evaluation*, the resultant context-aware models are evaluated with the associated contextual data.

Nowadays, machine learning classification techniques have been successfully used while building data-driven intelligent systems in various application areas including smartphone apps [3–5]. In the area of machine learning, a tree-like model is one of the most popular approaches for predicting context-aware smartphone usage [6, 7]. In this study, various popular *machine learning classification* techniques such as decision tree (DT), random forest (RF), k-nearest neighbor (KNN), support vector machines (SVM), naive Bayes classifier (NB), and deep learning by constructing artificial neural network (ANN) of multiple hidden layers, are used for creating models. Based on the context

pre-modeling tasks discussed above and these classification methods, we experimentally analyze personalized smartphone usage behavior utilizing their smartphone datasets. For this, we have first collected contextual apps usage datasets consisting of different categories of apps usages in different contexts that include both the user-centric context and device-centric context form individual smartphone users. We then analyze personalized apps usage behavior by exploring context pre-modeling tasks and various popular machine learning classification methods, as well as deep learning by constructing multiple hidden layers based neural networks. The effectiveness of these models is examined by considering prediction accuracy in terms of precision, recall, f-score, and ROC values, and make an empirical discussion within the scope of our study. Overall, this research aims to determine the *combination of context pre-modeling and classification methods* that work well together to build user-centric context-aware predictive models and systems for unseen test cases, to intelligently assist the users.

The rest of the paper is organized as follows. "The motivation and scope of the study" section motivates the work of context pre-modeling for building context-aware models and finds the scope of our study. "Background and related work" section provides background and related work in the scope of our study. "Context pre-modeling strategies" section provides an overview of various context pre-modeling strategies that are taken into account in our analysis. In "Implementing machine learning classification methods" section, we discuss various machine learning classification techniques to build a context-aware model. We have shown the experimental results in "Experimental results and evaluation" section. Some key observations including several application areas are summarized in "Discussion" section. Finally "Conclusion and future work" section concludes this paper and highlights the future work.

The motivation and scope of the study

In this section, our goal is to motivate the study of exploring context pre-modeling and classification methods that work well together in user-centric context-aware predictive models and applications in today's interconnected world, especially in the environment of IoT and smartphones. Hence, we also present the scope of our study.

We are currently living in the era of Data Science, Artificial Intelligence (AI), Internet-of-Things (IoT), and Cybersecurity, that are commonly known as the most popular latest technologies in *fourth industrial revolution* (4IR) [8, 9]. The computing devices like smartphones and corresponding applications are now used beyond the desktop, in diverse environments, and this trend toward ubiquitous and context-aware smart computing is accelerating. One key challenge that remains in this emerging research domain is the ability to enhance the behavior of any application by informing it of the surrounding contextual information such as temporal context, spatial context, social or device-related context, etc. Typically, by context, we refer to any information that characterizes a situation related to the interaction between humans, applications, and the surrounding environment [1, 10]. In the area of artificial intelligence, machine learning (ML) techniques can be used to build data-driven intelligent mobile systems based on these contextual information [3, 5, 7, 11]. However, contextual data needs to fed effectively to a machine learning model, so that the model can understand the contexts and behave intelligently. Thus, the overall performance of the machine learning-based context-aware

systems depends on the nature of the contextual data, and *context pre-modeling* tasks that can play a significant role to build an effective model, in which we are interested in this paper. Thus, context pre-modeling mainly manipulates contextual data to create features that make machine learning classification algorithms work well. Overall, the reasons for context pre-modeling tasks in a context-aware model and system can be summarized as below -

- To deal with a large number of contexts for reducing context-dimensions according to their importance, in order to generalize and simplify the model and makes it easy to interpret.
- To minimize over-fitting of a machine learning-based context-aware model and to improve the model's performance in terms of accuracy for unseen test cases.
- To reduce the context space and storage requirements as well as model complexity and computation cost in a context-aware model.
- To enable machine learning algorithms to train faster.

In this study, we mainly explore several context pre-modeling and classification methods that are relevant within the scope of this study, especially which are relevant in user-centric context-aware predictive modeling. Thus, we intend to tackle the following research questions with our study:

RQ1. Are the contexts varied with significant information related to the target behavioral activity classes of the users, on which several contextual features are determined, to build an effective data-driven context-aware predictive model?

RQ2. Do the machine learning classification based user-centric context-aware predictive models have an impact on context pre-modeling tasks, to effectively predict for unseen context-aware test cases?

To answer these research questions, in this work, we take into account an empirical analysis of several context pre-modeling tasks mentioned earlier, and to build user-centric context-aware predictive models based on various popular machine learning classification techniques. Users contextual apps usage datasets consisting of different categories of apps usages in different contexts are used in our empirical analysis.

Background and related work

Classification techniques are well-known and popular in the area of machine learning and data science, to build the prediction models. In general, the goal of classification is to effectively classify or predict the target class labels whose contextual features values are known in a context-aware model, but class values are unknown [12]. Several machine learning classification techniques such as ZeroR, naive Bayes, decision tree, random forest, k-nearest neighbor, support vector machines, artificial neural network, etc. exist with the capability of building context-aware predictive models in the domain of smartphone data analytics and to analyze context-aware personalized behavioral model [7].

In addition to classification techniques, unsupervised learning like clustering and association analysis, are well-known branches in the area of machine learning and data science and can be used for smartphone analytics. For instance, a number of authors use association learning [13–16], and clustering approaches [17, 18] for user

behavioral analytics based on contextual information. Although, several context pre-modeling tasks can help to build effective context-aware models based on these unsupervised approaches, in this work, we particularly focus on the supervised classification techniques for the purpose of building user-centric context-aware predictive models. Classification learning techniques typically build a context-aware model utilizing a given training dataset with contextual information and then the resultant predictive model can be used for testing purposes.

Among the traditional machine learning classification approaches, a tree-based, particularly, a decision tree based context-aware model is more effective to analyze user behavior in the domain of smartphone data analytics [7]. A number of researchers use decision tree classification technique in their study for different purposes [19–23]. For instance, Hong et al. [20], Lee et al. [21] propose context-aware model for providing personalized services utilizing context history. In [19], Zulkernain et al. design a rule-based context-aware system to intelligently assist mobile phone users. A decision tree based robust user behavior model utilizing contextual smartphone data has been presented in [11]. In addition to smartphone usage, decision tree based model can also be used in the domain of IoT or cybersecurity analytics [23]. Several decision tree learning approaches such as ID3 decision tree [24], C4.5 decision tree [25], behavioral decision tree BehavDT [6] exist with the capability of constructing contextual decision trees and building context-aware predictive models. Recently, Sarker et al. use an ensemble learning approach consisting of multiple decision trees for analyzing smartphone data [3]. Similarly, a number of researchers use ensemble learning approach, particularly, random forest classification technique in the area of context-aware mobile services in different purposes [26–28].

In addition to the context-aware tree-based models discussed above, several other machine learning classification techniques are used in the area of context-aware computing and smartphone analytics. For instance, Bozanta et al. [29] use the k-nearest neighbor classification technique while developing a contextually personalized recommender system. Ayu et al. use k-nearest neighbor classification in their study while recognize activity using mobile phone data. In [30], the authors use the k-nearest neighbor classifier while designing their recommendation system. Similarly, the naive Bayes classification technique is used in Ayu et al. [31], Fisher et al. [32] in their analysis. To analyze contextual mobile phone data, Pielot et al. [33], Bedogni et al. [27], Bayat et al. [34] have used support vector machines in their context-aware analysis to build context-aware models. In addition to these classical machine learning classification techniques, several research has been done based on artificial neural network [7, 35–37]. Although a number of research summarized above, has been done in the area of context-aware analysis and predictive modeling, more attention is needed to explore context engineering or pre-modeling tasks, to make the model more effective with reduced computational cost for unseen test cases, in which we are interested.

Therefore, in this paper, we mainly focus on *context pre-modeling* tasks and explore the most popular machine learning classification techniques to build effective user-centric context-aware predictive models utilizing multi-dimensional contextual data.

Context pre-modeling strategies

Context pre-modeling can also be considered as context engineering or pre-processing tasks such as context vectorization, context generation and extraction, context selection, and eventually context evaluation, etc. In the following, we briefly discuss these tasks to build effective context-aware models utilizing multi-dimensional contextual data.

Context vectorization

Context vectorization typically defines a good numerical measure to characterize a categorical context value. The vectors rather than direct contexts are used widely in machine learning because of the effectiveness and practicality of representing objects in a numerical way to help with many kinds of analyses. Thus, both the context transformation and context normalization are used to represent the contextual values into numerical values to feed into the machine learning technique.

Context transformation

Smartphone contextual data with users' behavioral activities, may contain categorical variables [5, 38]. These variables are typically stored as text values such as location at home, on the way; or Internet connectivity is on/off. Since machine learning classification algorithms are based on mathematical computing and relevant equations, it would cause a problem with such type of categorical values of the contexts. Thus, context transformation is needed for further processing, if the algorithms do not support categorical values. One hot encoding and label encoding are the most popular approaches to encode the categorical contextual features into numeric vectors [3, 39]. In one hot encoding technique, a significant number of dummy variable increases, and consequently increases the dimensions of the datasets. On the other hand, in label encoding, the context values converted directly into particular numeric values, and thus the number of features remains the same. In our experiments, we take into account the label encoding technique in the scope of our context-aware analysis. In this encoding technique, each categorical value is assigned a numeric value from 1 through N, where N represents the number of categories for a particular context. For instance, in terms of the spatial context, label encoding can turn users' diverse locations [at home, at the office, on the way, at home, on the way, at playground] into numeric values [0, 1, 2, 0, 2, 3].

Context normalization and scaling

The contextual dataset may contain context values highly varying in magnitudes, units, and range. As a result, further processing and building machine learning classification models may face problems while computations. With few exceptions, machine learning algorithms don't perform well when the input numerical attributes have very different scales [39]. Thus, context normalization or scaling is needed to resolve the issues involved. Context normalization is typically a method used to standardize the range of independent variables representing the contextual information. In data processing, it is also known as data scaling or standardization and is generally performed

during the data pre-processing step [12]. Standard scaler, Min-max scaler, Robust scaler, are the well-known normalization approaches in data pre-processing [39]. In our experiment, we take into account the Standard scaler by assuming the contextual data is normally distributed within each contextual feature and make scale them such that the distribution is centered around 0, with a standard deviation of 1. The mean and standard deviation are calculated for the contextual features based on the equations given below [39]:

$$Standardization : z = \frac{x - \mu}{\sigma} \tag{1}$$

$$Mean : \mu = \frac{1}{n} \sum_{i=1}^{n} x_i \tag{2}$$

$$Standard\ deviation : \sigma = \sqrt{\frac{1}{n} \sum_{i=1}^{n} (x_i - \mu)^2} \tag{3}$$

Thus, context transformation with normalization and scaling can be a good fit to represent contextual data into numeric values, to feed the necessary contextual information into the machine learning techniques, to build a data-driven context-aware model.

Contextual feature generation and extraction

Feature generation and extraction techniques typically provide a better understanding of the data, a way of improving prediction accuracy, as well as reducing computational cost or training time in a machine learning-based model or system [40]. Feature extraction typically is a process of combining existing features to produce a more useful one, also known as dimensionality reduction by which an initial set of raw data is reduced to more manageable groups for further processing [39]. For example, principal components analysis (PCA) can be used to extract a lower-dimensional space [12]. The key principle is that the extraction process takes into account creating brand new components based on the contextual information in the datasets. Several methods such as Principal Component Analysis (PCA), Singular Value Decomposition (SVD), and Linear Discriminant Analysis (LDA), etc. can be used for analyzing the significance of the contextual features [12, 39]. In our experiment, we take into account the PCA method, which is a popular and well-known feature extraction method used in the area of data science and machine learning. PCA method can produce new brand features or components by analyzing the characteristics of the contextual datasets [4]. Technically, PCA finds the eigenvectors of a covariance matrix with the highest eigenvalues and then uses those to project the data into a new subspace of equal or fewer dimensions. The below mathematical equations are relevant to the principal component analysis [4, 12].

$$var\ (U_1) = var\left(w^T X\right) = w^T Sw \tag{4}$$

$$r(X, Y) = \frac{\sum_{i=1}^{n}(X_i - \bar{X})(Y_i - \bar{Y})}{\sqrt{\sum_{i=1}^{n}(X_i - \bar{X})^2}\sqrt{\sum_{i=1}^{n}(Y_i - \bar{Y})^2}} \tag{5}$$

$$\mathrm{w}^T \mathrm{Sw} = \alpha_1 \mathrm{w}^T \mathrm{w} = \alpha_1 \tag{6}$$

$$\mathrm{w}^T \mathrm{w} = 1 \tag{7}$$

Where, the first principal component be a linear combination of X defined by coefficients or weights $w = [w_1...w_n]$, and can be written in matrix form as $\mathrm{U}_1 = \mathrm{w}^T \mathrm{X}$. Thus, in the scope of our study, we create principal components and calculate the variance of the components using this PCA method, considering the contexts and target behavioral activity of the users, to determine the significance of the contexts in a given dataset.

Contextual feature selection

Contextual feature selection could be another way to resolve the issues of high dimensional contextual data, as well as to analyze the significance of the contextual features in a predictive model. It selects the most useful features to train on among existing features [39]. Feature selection techniques typically provide a better understanding of the data, a way of improving prediction accuracy, as well as reducing computational cost or training time in a machine learning-based model or system [40]. A contextual dataset may contain data with high dimensions, and some of them might not contain significant information for building a machine learning classification-based model. Moreover, further processing with all the given features or attributes using machine learning techniques might give poor prediction results because of the over-fitting problem [23, 41]. Thus, an optimal number of features by selecting contexts, is needed not only to reduce the computational cost but also to build a more effective context-aware model with a higher accuracy rate. For this, we can filter less significant, irrelevant or redundant context from the given dataset, by analyzing the data patterns and dependency. Several statistical methods such as chi-squared test, analysis of variance test, correlation coefficient analysis, etc. can be used for analyzing the significance of the given features [12, 40]. In our experiment, we take into account the correlation of the contexts, known as Pearson correlation coefficient [12], which is the most popular method for analyzing how each context is correlated with each other and with the target behavioral activity of the users. Correlation is a well-known similarity measures between two contextual features, and measures the strength between features and with behavioral activity class. The correlation-based feature selection is based on the following hypothesis: "Good feature subsets contain features highly correlated with the target class, yet uncorrelated or less correlated to each other". If X and Y represent two random contextual variables, then the correlation coefficient between X and Y is defined as [12]

$$r(X, Y) = \frac{\sum_{i=1}^{n}(X_i - \bar{X})(Y_i - \bar{Y})}{\sqrt{\sum_{i=1}^{n}(X_i - \bar{X})^2}\sqrt{\sum_{i=1}^{n}(Y_i - \bar{Y})^2}} \tag{8}$$

Sarker *et al. J Big Data*   (2020) 7:51

Page 9 of 23

In the field of statistics, the formula Eq. 8 is often used to determine how strong that relationship is between those two variables X and Y with values between -1 and 1 considering both positive and negative correlation. Thus, in the scope of our study, we calculate the correlation coefficient values of each context and target behavioral activity of the users, to determine the significance of the contexts in a given dataset.

Implementing machine learning classification methods

In this section, we discuss the machine learning classification techniques to build context-aware predictive models, within the scope of our study. For this, we use the most popular machine learning library scikit-learn [42], as well as the deep learning libraries Keras [43], and Tensorflow [44] written in Python, where Keras is a high-level API that runs on top of TensorFlow [39]. In the following, we discuss the implemented machine learning classification techniques that are taken into account in our context-aware models.

Naïve Bayesian classification

Naïve Bayesian (NB) [45] classification is one of the popular supervised learning techniques based on statistical probability. This classifier is widely used in various application areas because of its simplicity and easy to build. In many application areas, it gives significant prediction results with the test cases. A naive Bayesian contextual model is based on the most popular Bayes' theorem in statistics, with the assumptions of independence between contextual features or predictors. Bayes' theorem typically provides a way of calculating the posterior probability of an event and is stated mathematically as the following equation [12]:
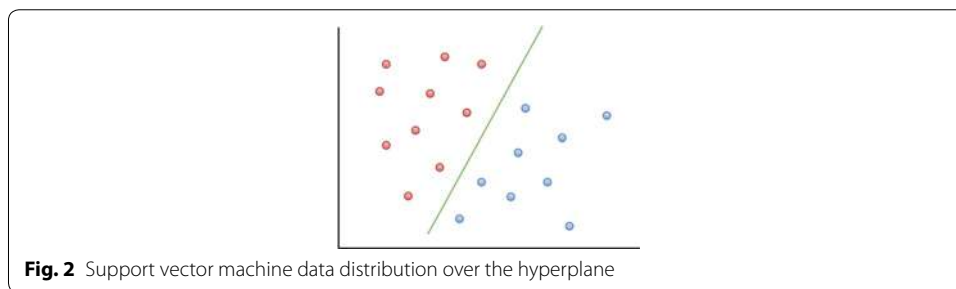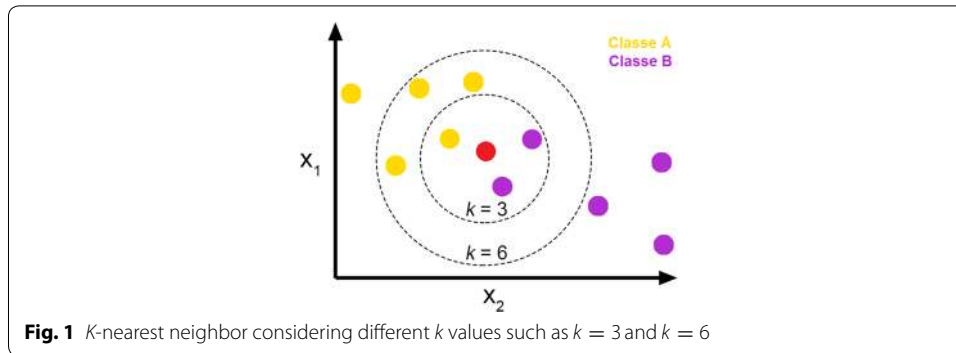
$$P(c|X) = \frac{P(X|c)P(c)}{P(X)} \tag{9}$$

$$P(c|x_1, x_2, ..., x_n) = \frac{P(x_1|c)P(x_2|c)...P(x_n|c)P(c)}{P(x_1)P(x_2)...P(x_n)} \tag{10}$$

where, c is a class variable that represents user behavioral activity and $P(c)$ is the class prior probability. $X = \{x_1, x_2, ..., x_n\}$ is a dependent feature vector of size n consists of contextual information, and $P(X)$ is the prior probability of a contextual feature. $P(c|X)$ represents the posterior probability of target class of the given contextual feature, while $P(X|c)$ is the likelihood which is the probability of contextual feature of a given class.

K-nearest neighbor

Another classification algorithm, K-nearest neighbors (KNN) [46] is a simple and popular technique, in the area of machine learning, used in many application areas for analyzing the prediction problems. This is also called as a lazy learning algorithm because of its working procedure. This technique does not have a specialized training phase for building a model rather uses all the data instances while classification. It classifies new test cases based on a 'feature similarity' measure, such as a distance function, e.g., Euclidean distance [12]. A case is classified by a majority voting of its neighbors indicating as k values. Figure 1 shows an example to understand the concept of k in a KNN algorithm,

Fig. 1 *K*-nearest neighbor considering different *k* values such as $k = 3$ and $k = 6$

Fig. 2 Support vector machine data distribution over the hyperplane

considering different k values, such as $k = 3$ and $k = 6$. While building our context-aware model, we take into $k = 5$, as the number of neighbors.

Support vector machine

Support vector machine (SVM) [47] classification is another popular supervised learning technique in machine learning, that can be used to build a context-aware predictive model. The computation concept behind this classifier is to find a hyperplane between the contextual data space, which best divides the dataset into two behavioral activity classes as shown in Fig. 2. The main principle to estimate the hyperplane is that it maximizes the margin between the two classes, and the vectors (cases) that define the hyperplane are the support vectors [12]. Overall, it works in two steps that include identifying the optimal hyperplane in contextual data space, and then to map the data instances according to the decision boundaries specified by the hyperplane for a given dataset.

In our context-aware model, we take into account several parameters including the kernel, while building the SVM based model. In the area of machine learning, kernel functions can be different types such as linear, poly, RBF, sigmoid, precomputed etc. [12, 39]. We use the Radial Basis Function (RBF), which is a popular kernel function used in various kernelized learning algorithms. In addition, the regularization parameter *C*, also known as the penalty parameter of the error term, which controls the trades-off the correct classification of training examples against the maximization of the decision function's margin. The strength of the regularization is inversely proportional to *C*. Thus we use $C = 1.0$, considering the trade-off between achieving a low training error, and a low testing error in our SVM based context-aware model.
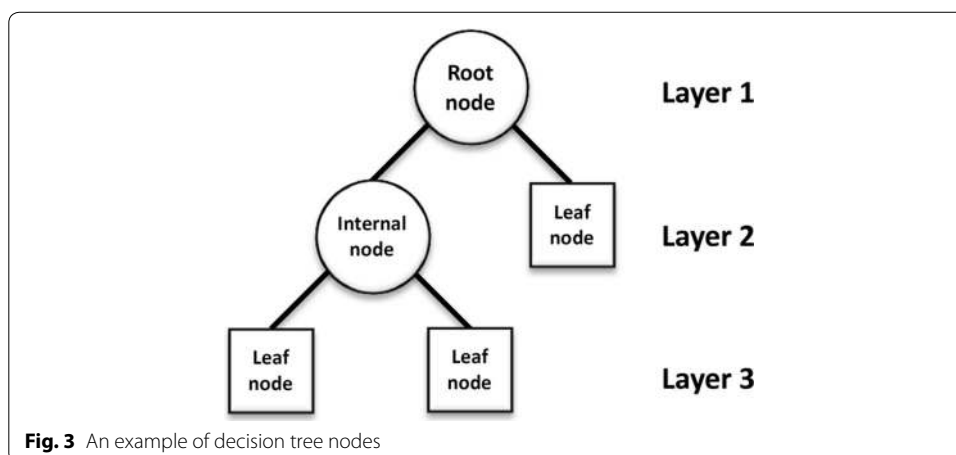
Decision tree

Decision tree (DT) [25] classification is a well-known supervised learning method, which is used widely for solving various prediction or classification problems in various application areas. A decision tree is a non-parametric supervised learning method that builds classification models in the form of a tree structure consisting of several nodes. It breaks down a contextual dataset into subsets and the associated tree is incrementally developed. It calculates entropy and information gain [25] while building the tree. In addition to entropy, "gini" function could be effective in a decision tree model [39]. An example of the decision tree nodes is shown in Fig. 3.

In our context-aware model, we take into account the best split at each node and use the "gini" function to measure the quality of a split in a decision tree model. In our decision tree model, nodes are expanded until all leaves are pure or until all leaves contain less than the minimum number of samples required to split an internal node, which is set as 2 in this experiment. We also set the minimum number of samples required to be at a leaf node as one. The maximum number of features are taken into account as the equal number of the contextual features in this experiment, to build our context-aware model.
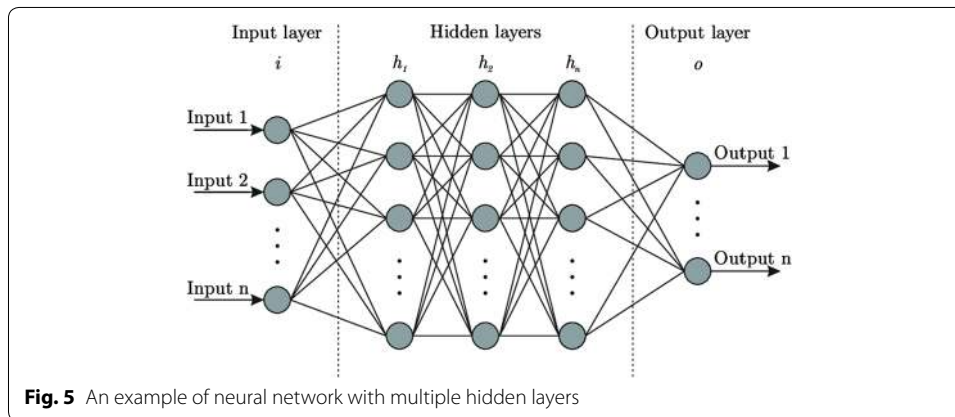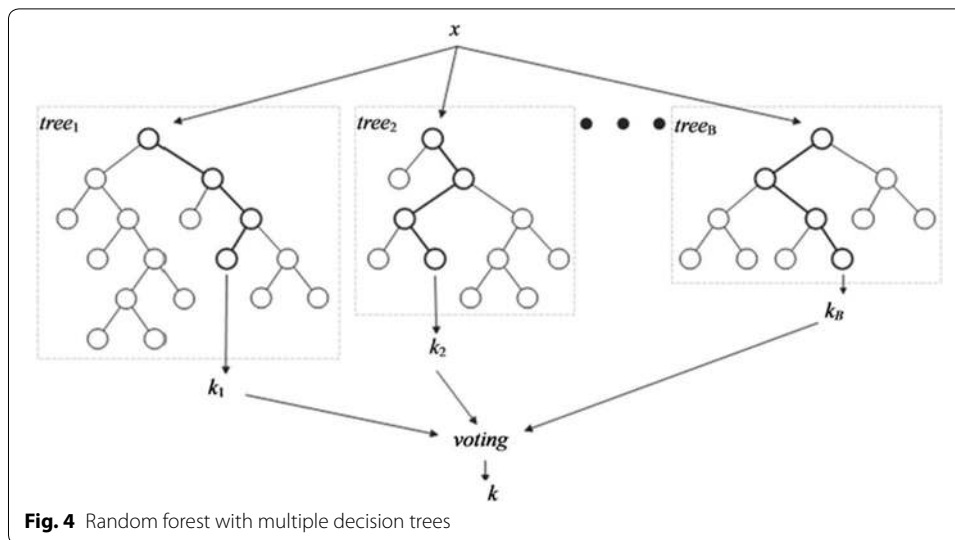
Random forest

The random forest (RF) [48] is an ensemble classification technique in machine learning consisting of multiple decision trees. It combines bootstrap aggregation (bagging) [49] and random feature selection [50] to construct a collection of decision trees exhibiting controlled variation. While training, each tree in a random forest contextual model learns from a random set of contextual data instances. The instances are drawn with replacement known as bootstrapping, which means some instances are used multiple times to build the tree. Overall, the entire random forest learner might have lower variance while building a model. After constructing the forest model, the prediction result is measured by taking into account the majority voting of the generated decision trees. Fig. 4 shows an example of a random forest structure considering several decision trees.

In our context-aware model, we take into account 100 trees while constructing the random forest. Although both the "gini" and "entropy" are the popular measures of

Fig. 3 An example of decision tree nodes

Sarker *et al. J Big Data*   (2020) 7:51

Page 12 of 23

Fig. 4 Random forest with multiple decision trees

Fig. 5 An example of neural network with multiple hidden layers

impurity of a node [12, 39], we use "gini" function that represents the average gain of purity by splits of a given context to measure the quality of a split. We do not restrict the maximum depth while generating a tree, as we have not a huge number of contexts in this experiment. Thus nodes are expanded until all leaves are pure or until all leaves contain less than the minimum number of samples required to split an internal node, which is set as 2 in this experiment, mentioned in decision tree model as well. We also set the minimum number of samples required to be at a leaf node as one. The maximum number of features are taken into account as the square root of the total contextual features of this experiment, to build our context-aware model.

Artificial neural network

An artificial neural network (ANN) is mainly used for deep learning models. An ANN is comprised of a network of artificial neurons, also known as nodes of the network [12]. The nodes are connected by links and each link is associated with weight and they interact with each other in different layers. In this work, we consider a feed-forward artificial neural network consisting of several layers, such as the input layer, hidden layer,

and output layer [12, 39]. In our ANN model, the size of the input layer has been chosen according to the number of selected contextual features, and the number of neurons in the output layer is equal to the number of classes. The three hidden layers with the neurons 100 have been carefully selected to build our contextual neural network model, as shown in Fig. 5.

In our context-aware model, we take into account three hidden layers with 100 neurons and compile the neural network with Adam optimizer [39]. While training the network, we use 1000 epochs with the batch size 200. We also use a small value of 0.001 as the learning rate as it allows the model to reach the global minimum. Regarding the activation function, we use the Rectified Linear Unit (ReLU) that overcomes the vanishing gradient problem, as well as allows the model to learn faster and perform better compared with other activation functions like the Sigmoid [39, 51]. We empirically set these hyperparameters to build our context-aware model using the artificial neural network.

Experimental results and evaluation

In this section, we first describe the datasets including contexts, and apps usage, and then highlight the evaluation metrics that are taken into account to measure the effectiveness of various machine learning classification models. We finally discuss the experimental results in various dimensions related to our analysis.

Datasets

We have collected individuals' smartphone usage datasets consisting of different categories of smartphone apps such as social networking, instant messaging, mobile communications, entertainment, or other apps related to users' daily life services in different contexts. The contexts are—temporal context such as time-of-the-day [24-h-a-day], days-of-the-week [7-days-a-week]; spatial context or user location such as at home, at the office, at the canteen, in the playground, on the way, etc; user work status such as workday or holiday; user diverse mood such as normal, happy, or sad; user device-related context such as battery level low, medium, or full; phone profile such as phone notification setting general, silent, or vibration; user Internet connectivity such as WiFi connectivity on or off. Different types of apps are Facebook, Gmail, LinkedIn, Instagram, Youtube video, Live Sport, Whatsapp, Internet browsing, watching movies, Skype, listening musics, reading news, playing games, etc. Datasets are collected from June 2018 to October 2018 from several participants for experimental purposes.

Evaluation metric

To evaluate the machine learning-based models, we employ the most popular *K*-fold cross-validation technique in machine learning [12]. In our evaluation, we use $K = 10$ for generating train and test data to build a model, and measure the predicted accuracy that are defined as below:

- Precision: It measures the ratio between the number of apps usage behaviors that are correctly predicted and the total number of apps that are predicted. If TP and FP denote true positives and false positives then the formal definition of precision is [52]:

$$Precision = \frac{TP}{TP + FP} \tag{11}$$

- Recall: It measures the ratio between the number of apps usage behaviors that are correctly predicted and the total number of relevant apps. If TP and FN denote true positives and false negatives then the formal definition of recall is [52]:

$$Recall = \frac{TP}{TP + FN} \tag{12}$$

- F_1 score: It is a measure that combines both the precision and recall defined above. It represents the harmonic mean of precision and recall. The formal definition of F_1 score is [52]:

$$F_1\ score = 2 * \frac{Precision * Recall}{Precision + Recall} \tag{13}$$

- ROC value: Receiver Operating Characteristic (ROC) that summarizes the trade-off between true positive rate and false-positive rate for a machine learning-based predictive model [52].
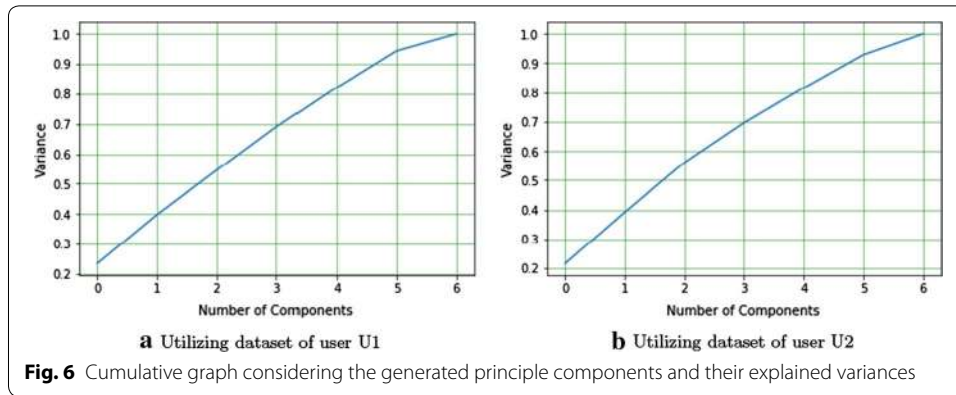
Context vectorization

As discussed above, the context values are category datatype that also considered as object type. Thus, we convert the context values into vectors to feed values into machine learning-based models. To achieve this goal, we first transform the context into numeric values. To do this, we use Label Encoder that transforms the context values into the desired numeric values. For instance, Internet connectivity on and off is transformed into 0 and 1. For more diverse values of contexts more numeric values are created. After performing encoding, we normalize the values using the Standard Scaler so that its distribution will have a mean value 0 and a standard deviation of 1. Table 1 shows the normalization value for some randomly selected instances of the dataset of user U1. Numerical values of different contexts shown in Table 1 are used to feed into machine learning-based models.

Prediction results of feature generation and extraction based model

In this experiment, we show the outcome results of the principal component-based feature extraction model. For this, we first generate the principle components and their

Table 1 Context normalization utilizing the dataset of user U1

Instance	Con1	Con2	Con3	Con4	Con5	Con6	Con7
I_1	1.72776303	− 0.59938852	0.12437413	− 1.02323747	− 1.26171878	0.9962435	0.06997602
I_{50}	0.6963469	− 0.59938852	0.12437413	0.97729024	− 1.26171878	0.43511498	− 1.21824026
I_{100}	− 0.77342109	1.66836695	− 0.58342305	− 1.02323747	1.13253553	− 1.24827061	− 1.21824026
I_{500}	0.82527391	− 0.59938852	1.53996849	0.97729024	1.13253553	0.43511498	1.35819231
I_{1000}	0.7479177	− 0.59938852	1.53996849	0.97729024	− 0.06459162	0.9962435	− 1.21824026
I_{2000}	1.5472652	− 0.59938852	0.12437413	− 1.02323747	− 1.26171878	0.43511498	1.35819231

Fig. 6 Cumulative graph considering the generated principle components and their explained variances

variance values. Figure 6 shows the cumulative graph considering all the principle components and their explained variances utilizing the datasets of user U1 and U2 respectively. The results are shown considering the variance with each component representing as 0, 1, 2, ...6 in the x-axis, where the variance value is between 0 and 1. If we observe Fig. 6, we see that, for each component, the variance graph increases linearly up to value 0.9. That means, all these generated components associated contain significant information to modeling.

In Tables 2 and 3, we have also shown the prediction results of the resultant context-aware models using various machine learning classification techniques utilizing the datasets of both the users U1 and U2. The techniques are random forest (RF), decision tree (DT), k-nearest neighbor (KNN), naive Bayes (NB), support vector machine (SVM), and artificial neural network (ANN) that are discussed briefly in earlier section. The results are shown by varying the variance threshold of v which is 90%, 70%, and

Table 2 Prediction results considering different variance utilizing the dataset of user U1

Classifier	$v = 90\%$			$v = 70\%$			$v = 50\%$		
	Precision	Recall	F_1 Score	Precision	Recall	F_1 Score	Precision	Recall	F_1 Score
RF	0.91	0.91	0.90	0.90	0.90	0.90	0.86	0.86	0.86
DT	0.90	0.90	0.90	0.90	0.90	0.90	0.86	0.86	0.86
KNN	0.84	0.83	0.83	0.82	0.81	0.81	0.79	0.78	0.78
NB	0.67	0.72	0.69	0.66	0.71	0.68	0.60	0.65	0.63
SVM	0.74	0.75	0.74	0.73	0.74	0.73	0.70	0.70	0.70
ANN	0.81	0.80	0.80	0.79	0.77	0.78	0.39	0.38	0.36

Table 3 Prediction results considering different variance utilizing the dataset of user U2

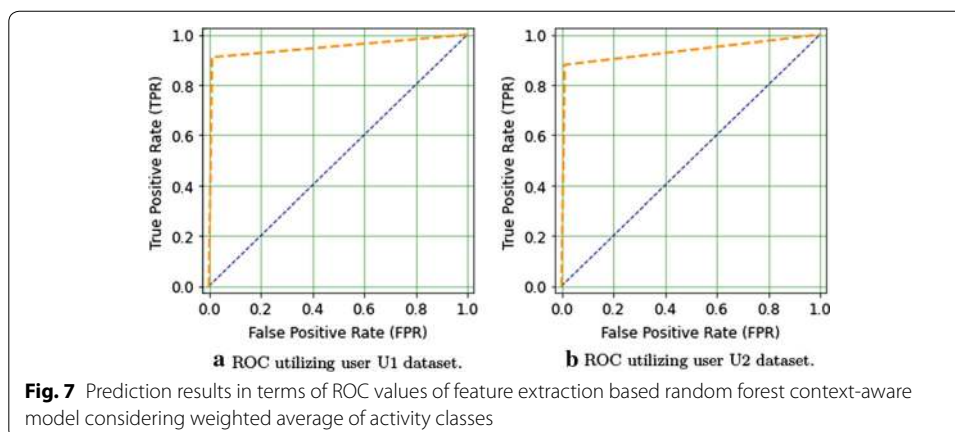| Classifier	$v = 90\%$			$v = 70\%$			$v = 50\%$		
	Precision	Recall	F_1 Score	Precision	Recall	F_1 Score	Precision	Recall	F_1 Score
RF	0.88	0.88	0.88	0.88	0.88	0.88	0.86	0.86	0.86
DT	0.88	0.87	0.87	0.87	0.87	0.87	0.86	0.86	0.86
KNN	0.85	0.84	0.84	0.85	0.84	0.84	0.79	0.78	0.78
NB	0.66	0.71	0.68	0.65	0.71	0.68	0.58	0.63	0.61
SVM	0.73	0.74	0.73	0.72	0.72	0.72	0.67	0.66	0.66
ANN	0.81	0.81	0.81	0.80	0.79	0.79	0.69	0.68	0.68

50%.If we observe Tables 2 and 3, we can see that different threshold values give different results for each machine learning-based model. For 50%, it decreases the prediction results in terms of precision, recall, and f-score. The reason is that low cumulative variance value as a threshold, may lose the context information and consequently low prediction results. However, a larger value of threshold increases the number of components in the model, consequently, increases the computational complexity and decreases the prediction results in several cases depending on the dataset characteristics. Thus, we choose 90% as a threshold as the variance graph increases linearly up to 0.9, to output the prediction results. For another dataset, this value might be changed depending on their data characteristics and patterns. According to these results shown in Tables 2 and 3, we can conclude that the random forest classification based context-aware model performs comparatively better than other classification techniques for a particular chosen threshold of variance. Besides, to these prediction results, we also show the ROC values considering the random forest classification model utilizing the datasets of both the users U1 and U2 in Fig. 7. The reason for getting better results with component-based random forest model is that this model fits multiple decision trees with the generated feature components, and averages all the single tree results.

Overall, we can conclude from our experimental results is that feature extraction by generating new components based context-aware models are capable to determine the significance of the created components. As a result, an effective context-aware model can be built up with this feature extraction based approach that is also able to provide the significant prediction results with the trade-off the components and the prediction accuracy according to the preferred variance in a model.

Prediction results of feature selection based model

In this experiment, we show the outcome results of the correlation-based feature selection approach for building context-aware models. For this, we first calculate the correlation of each context, and later with the target behavioral activity class as well. Tables 4 and 5 show the correlation values considering all the contexts utilizing the datasets of user U1 and U2 respectively. The results are shown considering the correlation with each context representing *Con*1, *Con*2, ..., *Con*7, where the value is between -1 and $+1$.

a ROC utilizing user U1 dataset.   **b** ROC utilizing user U2 dataset.

Fig. 7 Prediction results in terms of ROC values of feature extraction based random forest context-aware model considering weighted average of activity classes

Table 4 Correlation coefficient utilizing the dataset of user U1

	Con1	Con2	Con3	Con4	Con5	Con6	Con7
Con1	1.000000	− 0.605791	− 0.057758	− 0.080345	− 0.106224	− 0.000900	0.077178
Con2	− 0.605791	1.000000	0.029491	0.036674	0.040462	0.011684	− 0.038999
Con3	− 0.057758	0.029491	1.000000	− 0.002305	− 0.080622	− 0.006046	− 0.004714
Con4	− 0.080345	0.036674	− 0.002305	1.000000	− 0.059981	0.030254	0.049993
Con5	− 0.106224	0.040462	− 0.080622	− 0.059981	1.000000	− 0.069363	− 0.029213
Con6	− 0.000900	0.011684	− 0.006046	0.030254	− 0.069363	1.000000	− 0.004885
Con7	0.077178	− 0.038999	− 0.004714	0.049993	− 0.029213	− 0.004885	1.000000

Table 5 Correlation coefficient utilizing the dataset of user U2

	Con1	Con2	Con3	Con4	Con5	Con6	Con7
Con1	1.000000	− 0.477266	− 0.018101	0.070562	0.081548	− 0.004683	0.002113
Con2	− 0.477266	1.000000	− 0.052795	− 0.009941	− 0.068945	− 0.116782	− 0.007300
Con3	− 0.018101	− 0.052795	1.000000	0.079590	− 0.013037	0.032749	− 0.110511
Con4	0.070562	− 0.009941	0.079590	1.000000	0.096467	− 0.108341	− 0.127522
Con5	0.081548	− 0.068945	− 0.013037	0.096467	1.000000	− 0.134262	0.078804
Con6	− 0.004683	− 0.116782	0.032749	− 0.108341	− 0.134262	1.000000	− 0.014646
Con7	0.002113	− 0.007300	− 0.110511	− 0.127522	0.078804	− 0.014646	1.000000

Table 6 Correlation coefficient with target class utilizing the dataset of user U1 and U2

User U1		User U2	
Context	Target Class	Context	Target Class
Con1	− 0.041343	Con1	− 0.063652
Con2	0.086053	Con2	0.079523
Con3	0.016337	Con3	− 0.058442
Con4	− 0.010151	Con4	− 0.086163
Con5	− 0.035780	Con5	− 0.142989
Con6	− 0.035325	Con6	0.019234
Con7	− 0.003021	Con7	− 0.093992

If we observe Tables 4 and 5, we see that, for each context, it generates a particular correlation value according to their relevance with other contexts. The negative correlation shown in Tables 4 and 5 is a relationship between two contextual variables in which one variable increases as the other decreases, and vice versa. The higher value represents highly correlated with each other and vice-versa. For instance, a perfect positive correlation is represented by the value +1 while a perfect negative correlation is represented by the value -1. For any 0 value, it indicates no correlation. A good feature subset contains features with less correlated or even uncorrelated to each other. The reason is that features with high correlation are more linearly dependent, and hence have almost the same effect on the dependent target class variable. In addition to correlation values between the contextual features, Table 6 also shows the correlation values for each context with the target class variable for the user U1 and U2 respectively, where the value

Table 7 Prediction results considering different correlation utilizing the dataset of user U1
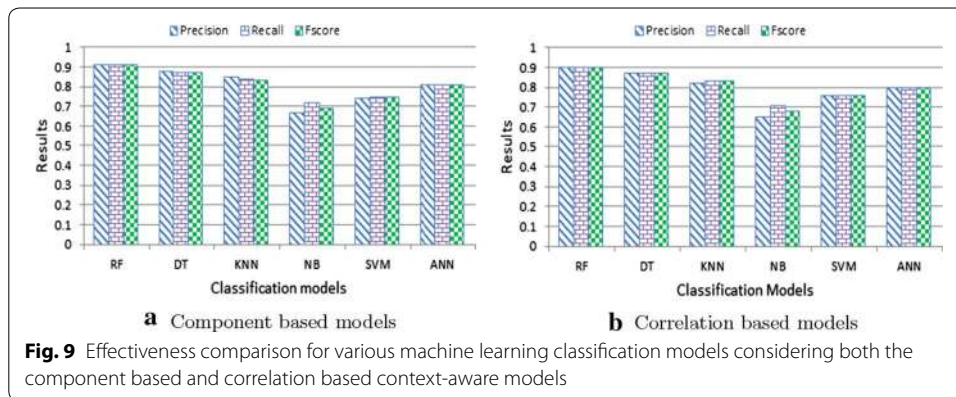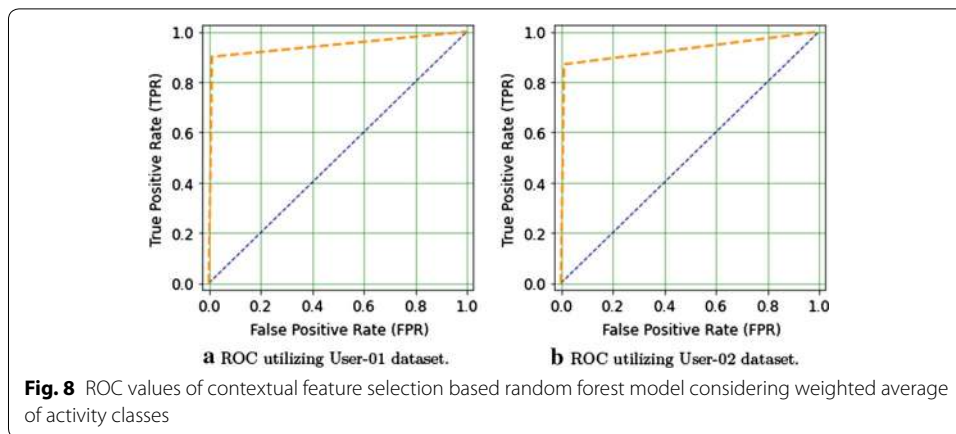
Classifier	t = 90%			t = 70%			t = 10%		
	Precision	Recall	F_1 Score	Precision	Recall	F_1 Score	Precision	Recall	F_1 Score
RF	0.90	0.90	0.90	0.90	0.90	0.90	0.31	0.30	0.29
DT	0.87	0.87	0.87	0.87	0.87	0.87	0.31	0.30	0.29
KNN	0.82	0.82	0.82	0.82	0.82	0.82	0.27	0.24	0.23
NB	0.66	0.72	0.69	0.65	0.71	0.68	0.60	0.64	0.62
SVM	0.73	0.74	0.74	0.73	0.73	0.73	0.70	0.70	0.70
ANN	0.79	0.79	0.79	0.79	0.79	0.79	0.21	0.20	0.19

Table 8 Prediction results considering different correlation utilizing the dataset of user U2

Classifier	t = 90%			t = 70%			t = 10%		
	Precision	Recall	F_1 Score	Precision	Recall	F_1 Score	Precision	Recall	F_1 Score
RF	0.88	0.87	0.87	0.88	0.87	0.87	0.62	0.61	0.61
DT	0.88	0.87	0.87	0.88	0.87	0.87	0.62	0.60	0.61
KNN	0.85	0.85	0.85	0.85	0.85	0.85	0.63	0.62	0.62
NB	0.66	0.71	0.68	0.65	0.71	0.68	0.58	0.63	0.61
SVM	0.73	0.74	0.73	0.72	0.72	0.72	0.67	0.66	0.66
ANN	0.80	0.79	0.79	0.80	0.79	0.79	0.24	0.24	0.22

is also between -1 and $+1$. In this case, a good feature subset contains features that are highly correlated with the target class variable for classification, as the target class is directly influenced by these features according to their correlation values.

To show the effect of feature subsets selection with their correlation values, we have also shown the prediction results of the resultant context-aware models using various machine learning classification techniques, such as random forest (RF), decision tree (DT), k-nearest neighbor (KNN), naive Bayes (NB), support vector machine (SVM), and artificial neural network (ANN), utilizing the datasets of both the users U1 and U2. The results are shown by varying the correlation threshold t with 90%, 70% and 10% in Tables 7 and 8 respectively. If we observe Tables 7 and 8, we can see that different threshold values give different results for each machine learning-based model. For 10%, it decreases the prediction results in terms of precision, recall, and f-score. The reason is that low correlation value as a threshold, may lose the context information and consequently low prediction results. However, a larger value of threshold increases the number of contexts in the model, consequently, increases the computational complexity and decreases the prediction results in several cases depending on the dataset characteristics. Thus, we choose 90% as a threshold as the correlation threshold, to output the prediction results. For another dataset, this value might be changed depending on their data characteristics and patterns. According to these results shown in Tables 7 and 8, we can conclude that the random forest classification based context-aware model performs comparatively better than other classification techniques for a particular chosen threshold of correlation. Besides these prediction results, we also show the ROC values in Fig. 8, considering the random forest classification model utilizing the datasets of both the users U1 and U2. The reason for getting better results with correlation-based

Fig. 8 ROC values of contextual feature selection based random forest model considering weighted average of activity classes

Fig. 9 Effectiveness comparison for various machine learning classification models considering both the component based and correlation based context-aware models

random forest model is that this model fits multiple decision trees with the selected feature subsets, and averages all the single tree results.

Overall, we can conclude from our experimental results is that feature selection based context-aware models are capable to determine the significance of the selected features. As a result, an effective context-aware model can be built up with this feature selection based approach that is also able to provide the significant prediction results with the trade-off the selected features and the prediction accuracy according to the preferred correlation value in the model.

Effectiveness comparison

In this experiment, we compute and compare the effectiveness of both the principal component-based context-aware model, and correlation-based context-aware model, using above mentioned machine learning classification methods. These are random forest (RF), decision tree (DT), k-nearest neighbor (KNN), naive Bayes (NB), support vector machine (SVM), and artificial neural network (ANN) that are discussed briefly in earlier section. Figure 9a, b show the relative comparison of prediction results in terms of precision, recall, f-score utilizing a collection of apps usage datasets of ten individual participants.

If we observe Fig. 9a, b, we see that both the component-based models and the correlation-based models give significant prediction results for random forest learning comparing to other learners. To calculate these results, we consider 90% variance, and 90% correlation as the threshold value, to select the number of components, and contexts respectively. However, different values may give different results that are discussed earlier. The main difference between these two models are - in a component-based model the contexts are converted as principal components and are used in the model, and the number of components is chosen based on variance. On the other hand, a correlation-based method directly uses a subset of the contexts in the datasets, and the number of contexts are chosen according to their correlation. Overall, from Fig. 9a, b, we can conclude that for a particular chosen threshold of variance, or correlation, the random forest classification based context-aware model performs comparatively better than other classification techniques.

Discussion

According to our empirical analysis of context pre-modeling tasks on contextual datasets discussed in the earlier section, it can be concluded that the integration of context pre-modeling and machine learning classification methods work well, in user-centric context-aware predictive models. Moreover, our findings through experimental analysis have shown that each context pre-modeling task involved in this work, such as context vectorization by defining a good numerical measure through transformation and normalization, context generation and extraction by creating new brand principal components, context selection by taking into account a subset of contexts according to their correlations, has a particular role for building effective context-aware predictive models. Such context-aware models can be applied in various domains of today's interconnected world, especially in the environment of IoT and smartphones, such as smart cities, smart environments, home automation, eHealth, cybersecurity, and emergencies etc, where a number of contexts and data-driven services based on machine learning techniques are involved. Moreover, our analysis and discussion that we have done throughout the paper can also be helpful for the professionals of cybersecurity or mobile/IoT security domain, where high-dimension of security features are involved to build data-driven decision making [53].

Our findings show that the random forest classification based context-aware model performs comparatively better than other classification techniques, such as naive Bayes, decision tree, *k*-nearest neighbor, support vector machine, artificial neural network, etc. in both the component-based and correlation-based models for a particular chosen threshold of variance, or correlation discussed in our experiments. Based on our analysis, we can conclude that - the associated contexts varied with significant information related to the target user behavioral activity class, on which several contextual features are determined. Moreover, the machine learning classification based user-centric context-aware predictive models also have an impact on the mentioned context pre-modeling tasks for effectively predicting for unseen context-aware test cases. As we claim that considering higher dimensions of contexts may cause over-fitting problems, and consequently decrease the prediction accuracy, these methods could be effective to resolve the issues depending on their contextual data characteristics.

This study has its limitations. Although, our empirical analysis has been done on contextual data, however, the number of contexts in the datasets is limited. Beyond that, this empirical analysis might deliver limited insights concerning good context engineering when applied to only a few number of contexts. Our analysis would be more practical when more dimensions of contextual data available for building user-centric context-aware models. Although we have taken into account smartphone apps usages datasets as an example throughout this paper and done experiments accordingly, we believe that our analysis would beneficial for building context-aware predictive models in the area of relevant human-centric computing, applications, and services. The reason is that while discussing the benchmarking approach in context-aware modeling for a particular human-centric application, the question of how to determine whether a given context is significant or not was raised. Thus, we recommend an assessment of the available contexts through our empirical analysis either based on creating new components, or calculating correlations. Further research is needed to give concrete advice on which contextual indicators are the most relevant for a particular context-aware system, and then collecting or analyzing the contextual raw data from our surrounding dynamic environment accordingly.

Conclusion and future work

In this paper, we have presented an empirical analysis of context pre-modeling for building user-centric context-aware predictive models utilizing smartphone datasets. For this purpose, we have explored several context pre-modeling tasks, such as context vectorization that includes context transformation and normalization, context generation and extraction by creating new brand principal components, or context selection by taking into account a subset of contexts according to their correlations, and eventually context evaluation to build effective context-aware models utilizing multi-dimensional contextual data. For creating models, we have used various popular machine learning classification techniques such as decision tree, random forest, *k*-nearest neighbor, support vector machines, naive Bayes classifier, and deep learning by constructing an artificial neural network of multiple hidden layers. The effectiveness of these models is examined by considering prediction accuracy in terms of precision, recall, f-score, accuracy, and ROC values. Our analysis has shown that exploring context pre-modeling and classification methods work well together in user-centric context-aware predictive models and applications in today's interconnected world, especially in the environment of IoT and smartphones. We believe that this study would be helpful to application developers to build corresponding human-centric real-life applications for the end-users, particularly, where higher dimensions of contexts involved.

To assess the effectiveness of the discussed machine learning-based context-aware models by collecting more dimensions of contextual data in the domain of IoT services and security, and measure the effectiveness in application level could be a future work.

Author details
[1] Swinburne University of Technology, Melbourne, VIC 3122, Australia. [2] Chittagong University of Engineering and Technology, Chittagong 4349, Bangladesh. [3] Macquarie University, Sydney, NSW 2109, Australia. [4] Computer Science Department, Faculty of Computing and Information Technology, King Abdulaziz University, Jeddah 21589, Saudi Arabia. [5] School of Engineering and Sciences, Tecnologico de Monterrey, Av. E Garza Sada 2501, Monterrey, N.L, Mexico.

References
1. Sarker IH. Context-aware rule learning from smartphone data: survey, challenges and future directions. J Big Data. 2019;6(1):1–25.
2. Sarker IH. Mobile data science: Towards understanding data-driven intelligent mobile applications. EAI Endorsed Transactions on Scalable Information Systems. 2018;5(19). EAI.
3. Sarker IH, Salah K. Appspred: predicting context-aware smartphone apps using random forest learning. Internet Things. 2019;8:100106.
4. Sarker IH, Abushark YB, Khan AI. Contextpca: predicting context-aware smartphone apps usage based on machine learning techniques. Symmetry. 2020;12(4):499.
5. Srinivasan V, Moghaddam S, Mukherji A. Mobileminer: mining your frequent patterns on your phone. In: Proceedings of the international joint conference on pervasive and ubiquitous computing, Seattle, WA, USA, 13–17 September. New York, USA: ACM; 2014. p. 389–400.
6. Sarker IH, Colman A, Han J, Khan AI, Abushark YB, Salah K. Behavdt: a behavioral decision tree learning to build user-centric context-aware predictive model. Mob Netw Appl. 2019;25:1151–61.
7. Sarker IH, Kayes A, Watters P. Effectiveness analysis of machine learning classification models for predicting personalized context-aware smartphone usage. J Big Data. 2019;6(1):1–28.
8. Ślusarczyk B. Industry 4.0: Are we ready? Polish J Manage Stud. 2018;17:232–48.
9. Cao L. Data science: a comprehensive overview. ACM Comput Surv. 2017;50(3):43.
10. Dey AK. Understanding and using context. Pers Ubiquitous Comput. 2001;5(1):4–7.
11. Sarker IH. A machine learning based robust prediction model for real-life mobile phone data. Internet Things. 2019;5:180–93.
12. Han J, Pei J, Kamber M. Data mining: concepts and techniques. Amsterdam: Elsevier; 2011.
13. Zhu H, Chen E, Xiong H, Yu K, Cao H, Tian J. Mining mobile user preferences for personalized context-aware recommendation. ACM Trans Intell Syst Technol. 2014;5(4):58.
14. Sarker IH, Salim FD. Mining user behavioral rules from smartphone data through association analysis. In: Proceedings of the 22nd Pacific-Asia conference on knowledge discovery and data mining (PAKDD), Melbourne, Australia. Springer; 2018. p. 450–61.
15. Sarker IH, Colman A, Han J. Recencyminer: mining recency-based personalized behavior from contextual smartphone data. J Big Data. 2019;6(1):1–21.
16. Agrawal R, Srikant R. Fast algorithms for mining association rules. In: Proceedings of the international joint conference on very large data bases, Santiago Chile, vol. 1215. Springer; 1994. p. 487–99.
17. Sarker IH, Colman A, Kabir MA, Han J. Individualized time-series segmentation for mining mobile phone user behavior. Comput J. 2018;61(3):349–68.
18. Sarker IH. Research issues in mining user behavioral rules for context-aware intelligent mobile applications. Iran J Comput Sci. 2018;2:41–51.
19. Zulkernain S, Madiraju P, Ahamed SI, Stamm K. A mobile intelligent interruption management system. J UCS. 2010;16(15):2060–80.
20. Hong J, Suh E-H, Kim J, Kim S. Context-aware system for proactive personalized service based on context history. Expert Syst Appl. 2009;36(4):7448–57.
21. Lee W-P. Deploying personalized mobile services in an agent-based environment. Expert Syst Appl. 2007;32(4):1194–207.

22. Sarker IH, Kabir MA, Colman A, Han J. An effective call prediction model based on noisy mobile phone data. In: Proceedings of the 2017 ACM international joint conference on pervasive and ubiquitous computing and proceedings of the 2017 ACM international symposium on wearable computers, USA. ACM; 2017. p. 193–6.
23. Sarker IH, Abushark YB, Alsolami F, Khan AI. Intrudtree: a machine learning based cyber security intrusion detection model. Symmetry. 2020;12(5):754.
24. Quinlan JR. Induction of decision trees. Mach Learn. 1986;1(1):81–106.
25. Quinlan JR. C4.5: Programs for machine learning. Mach Learn; 1993.
26. Pielot M. Large-scale evaluation of call-availability prediction. In: Proceedings of the international joint conference on pervasive and ubiquitous computing. ACM; 2014. p. 933–7.
27. Bedogni L, Di Felice M, Bononi L. Context-aware android applications through transportation mode detection techniques. Wireless Commun Mob Comput. 2016;16(16):2523–41.
28. Turner LD, Allen SM, Whitaker RM. Interruptibility prediction for ubiquitous systems: conventions and new directions from a growing field. In: Proceedings of the 2015 ACM international joint conference on pervasive and ubiquitous computing. ACM; 2015. p. 801–12.
29. Bozanta A, Kutlu B. Developing a contextually personalized hybrid recommender system. Mob Inform Syst. 2018;. https://doi.org/10.1155/2018/3258916.
30. Middleton SE, Shadbolt NR, De Roure DC. Ontological user profiling in recommender systems. ACM Trans Inform Syst. 2004;22(1):54–88.
31. Ayu MA, Ismail SA, Matin AFA, Mantoro T. A comparison study of classifier algorithms for mobile-phone's accelerometer based activity recognition. Procedia Eng. 2012;41:224–9.
32. Fisher R, Simmons R. Smartphone interruptibility using density-weighted uncertainty sampling with reinforcement learning. In: 2011 10th international conference on machine learning and applications and workshops, vol. 1. IEEE; 2011. p. 436–41.
33. Pielot M, De Oliveira R, Kwak H, Oliver N. Didn't you see my message?: predicting attentiveness to mobile instant messages. In: Proceedings of the SIGCHI conference on human factors in computing systems. ACM; 2014. p. 3319–28.
34. Bayat A, Pomplun M, Tran DA. A study on human activity recognition using accelerometer data from smartphones. Procedia Comput Sci. 2014;34:450–7.
35. Alawnah S, Sagahyroon A. Modeling of smartphones' power using neural networks. EURASIP J Embedded Syst. 2017;2017(1):22.
36. Tan GW-H, Ooi K-B, Leong L-Y, Lin B. Predicting the drivers of behavioral intention to use mobile learning: A hybrid sem-neural networks approach. Comput Hum Behav. 2014;36:198–213.
37. Rajashekar D, Zincir-Heywood AN, Heywood MI. Smart phone user behaviour characterization based on autoencoders and self organizing maps. In: 2016 IEEE 16th international conference on data mining workshops (ICDMW). IEEE; 2016. p. 319–26.
38. Sarker IH. Behavminer: Mining user behaviors from mobile phone data for personalized services. In: Proceedings of the 2018 IEEE international conference on pervasive computing and communications (PerCom 2018), Athens, Greece. IEEE; 2018.
39. Géron A. Hands-on machine learning with scikit-learn, keras, and tensorflow: concepts, tools, and techniques to build intelligent systems. Newton: O'Reilly Media; 2019.
40. Liu H, Motoda H. Feature extraction, construction and selection: a data mining perspective. Berlin: Springer; 1998. p. 453.
41. Sneha N, Gangil T. Analysis of diabetes mellitus for early prediction using optimal features selection. J Big Data. 2019;6(1):13.
42. Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B, Grisel O, Blondel M, Prettenhofer P, Weiss R, Dubourg V, et al. Scikit-learn: machine learning in python. J Mach Learn Res. 2011;12:2825–30.
43. Ketkar N. Introduction to keras. Berkeley: Apress; 2017. p. 97–111.
44. Abadi M, Barham P, Chen J, Chen Z, Davis A, Dean J, Devin M, Ghemawat S, Irving G, Isard M, et al. Tensorflow: a system for large-scale machine learning. 2016;5:265–83.
45. John GH, Langley P. Estimating continuous distributions in bayesian classifiers. In: Proceedings of the eleventh conference on uncertainty in artificial intelligence. Morgan Kaufmann Publishers Inc.; 1995. p. 338–45.
46. Aha DW, Kibler D, Albert MK. Instance-based learning algorithms. Mach Learn. 1991;6(1):37–66.
47. Keerthi SS, Shevade SK, Bhattacharyya C, Murthy KRK. Improvements to platt's smo algorithm for svm classifier design. Neural Comput. 2001;13(3):637–49.
48. Breiman L. Random forests. Mach Learn. 2001;45(1):5–32.
49. Breiman L. Bagging predictors. Mach Learn. 1996;24(2):123–40.
50. Amit Y, Geman D. Shape quantization and recognition with randomized trees. Neural Comput. 1997;9(7):1545–88.
51. Cicceri G, De Vita F, Bruneo D, Merlino G, Puliafito A. A deep learning approach for pressure ulcer prevention using wearable computing. Human Centric Comput Inform Sci. 2020;10(1):5.
52. Witten IH, Frank E, Trigg LE, Hall MA, Holmes G, Cunningham SJ. Weka: practical machine learning tools and techniques with java implementations; 1999.
53. Sarker IH, Kayes ASM, Badsha S, et al. Cybersecurity data science: an overview from machine learning perspective. J Big Data. 2020;7:41. https://doi.org/10.1186/s40537-020-00318-5.

Publisher's Note