ISSN: 2454-9940

INTERNATIONAL JOURNAL OF APPLIED SCIENCE ENGINEERING AND MANAGEMENT

INTERNATIONAL JOURNAL OF APPLIED SCIENCE ENGINEERING AND MANAGEMENT

ISSN2454-9940www.ijasem.org

Vol 9, Issuse.1 Feb 2021

Patient Treatment Time Prediction Algorithm for Hospital Queuing-Recommendation in a Big Data Environment

B.Venkateswarlu[1],Erugu Krishna[2],Firdose Fathima[3],Dr.M.Sreenivasulu [4],

Abstract— ***One of the most difficult problems hospitals confront is effectively managing the patient queue in order to decrease patient wait times and congestion. Patients' irritation is exacerbated when they are forced to wait for lengthy periods of time for no good reason. The amount of time a patient must wait depends on how long the line behind him is. Patients would find it more convenient and preferred if they could get real-time information about expected wait times and the most efficient treatment plans through a mobile application. Because of this, we have developed a Patient Treatment Time Prediction (PTTP) method to estimate how long a patient will have to wait before receiving treatment. For each job, we develop a patient treatment time model based on real-world patient data collected from multiple hospitals. The treatment time for each patient in the current queue of each job is anticipated based on this large-scale, realistic dataset. A Hospital Queuing-Recommendation (HQR) system is created based on the estimated wait time. HQR determines the most cost-effective and time-saving treatment option for each patient. The PTTP algorithm and HQR system are required to respond quickly and efficiently because of the vast, realistic dataset and the need for real-time reaction. The National Supercomputing Center in Changsha (NSCC) uses an Apache Spark-based cloud solution to meet the aforementioned aims. Patients' wait times in hospitals may be reduced by recommending an appropriate treatment plan based on extensive testing and simulation findings.***

INTRODUCTION

Motivation

In most hospitals, there is now an overcrowding problem and no efficient way to manage the patient queues. Waiting time forecast for patients is a hard task since each patient may need a variety of procedures, such as a blood test or glucose level check or an ultrasound, throughout their treatment. Treatment tasks or tasks are used in this work to refer to each of these stages and processes. It is very difficult to forecast how long a certain therapy job will take for each individual patient, making time estimation and recommendation extremely difficult. According to their health, a patient is often obliged to undertake various types of exams, inspections, and tests (together referred to as chores). In this instance, each patient may be forced to do more than one activity. If one job is reliant, another may have to wait until that work is completed. Most patients are need to take medication. Hunan University's College of Computer Science and Electronic Engineering, together with the National Supercomputing Center in Changsha, Hunan, Changsha, 410082, China, is home to the research of Jianguo Chen, Ken Li, Zhu Tang, and Keqin Li. Kenli Li, lkl@hnu.edu.cn, is the author's point of contact. The COMSATS Institute of Information Technology, Islamabad, Pakistan, and Qatar University, Qatar are both home to Kashif Bilal. • Keqin Li is also affiliated with the State University of New York's Department of Computer Science in New Paltz, NY 12561, the United States. patiently wait in line for seemingly interminable lengths of time in order to get treatment. These guidelines are aimed at assisting hospitals in planning each queue of treatment chores, avoiding overcrowding, and ensuring that patients can finish their jobs on time and without interruption. To construct a patient treatment time consumption model, we draw on a wealth of real-world hospital data. Based on crucial characteristics such as patient treatment start time, finish time, patient age and detail treatment content for each separate job are assessed thoroughly and rigorously. Waiting durations vary dependent on a patient's health and the procedures they had during therapy. FIG. 1 depicts Fig. 1's patient treatment and wait process. Patient 1, Patient 2, and Patient 3 are shown in Fig. 1, along with the therapy activities that must be completed for each of them. Prior to X-rays, surgery or bandaging can't be carried out, for example.

Professor[1,2,3,4,] Assistant Professor[1,2,3,4], ,Associate Professor[1,2,3,4],
Department of CSE Engineering,
Pallavi Engineering College,
Mail ID:bvenkat1109@gmail.com, Mail ID:krishna.cseit@gmail.com,
Kuntloor(V),Hayathnagar(M),Hyderabad,R.R.Dist.-501505.

P atient1 must do tasks A, B, and D, however job D must wait until B is completed. As for P atient2 and P atient3, they need to complete assignments titled 'A-E-B-C' and 'D-E-C-A-B-C'. In addition, the number of patients in each task's queue varies, with 7 patients in task A's queue and 5 patients in task B's queue, for example. A PTTP (Patient Treatment Time Prediction) model is developed using historical hospital data in this article. In order to estimate the wait times for each treatment task, PTTP is used.

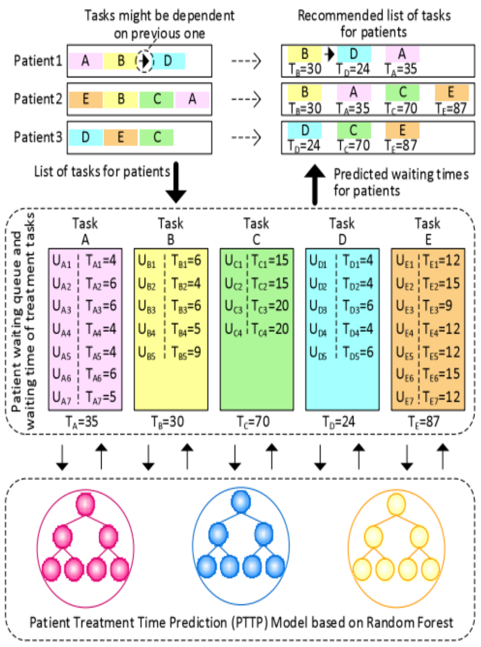

Tasks might be dependent on previous one
Recommended list of tasks for patients
Patient1
Patient2
Patient3
List of tasks for patients
Predicted waiting times for patients
Patient waiting queue and waiting time of treatment tasks
Task A
Task B
Task C
Task D
Task E
T_A=35
T_B=30
T_C=70
T_D=24
T_E=87
Patient Treatment Time Prediction (PTTP) Model based on Random Forest

Fig. 1. Workflow of patient treatment and wait model

This represents the total amount of time each patient has to wait in the current line. A Hospital Queuing-Recommendation (HQR) system then offers an efficient and easy treatment plan for each patient based on their specified treatment duties. In the waiting room, the trained PTTP model calculates how much time each patient will need for their treatment. Each task's current wait time may be calculated, such as TA = 35(min),TB = 30(min),TC = 70(min),TD = 24(min), and TE = 87(min). Finally, each patient's jobs are arranged according to the patient's waiting time, except for the dependent tasks. A suggestion for queuing is made for each patient, such as the suggested queuing 'A' for P atient1, 'B, A' for P atient2, and 'D, C' for P atient3' for each patient. Every task's waiting time is pre-calculated in real-time so that all necessary treatment may be completed in the quickest time possible. Queuing recommendations are updated in real time because of the dynamic nature of the task queues. This means that each patient may be counselled on how to go through his or her therapy in the smallest amount of time possible.

Our Contributions

PTTP and HQR systems are proposed in this study. Our system's real-time demands, massive data sets, and complexity necessitate the use of big data and cloud computing approaches. For each treatment task, the PTTP algorithm is trained using an enhanced Random Forest (RF) method, and the waiting time for each task is forecasted using the learned PTTP model. Finally, HQR makes a recommendation for a treatment method that is both effective and convenient for each patient. The proposed strategy and estimated waiting time may be seen in real time by patients utilising a mobile app. Extensive testing and results from actual applications indicate that the PTTP algorithm is very precise and fast. The following is a summary of our contributions to this work.

Based on the Random Forest (RF) algorithm, PTTP has been presented. The PTTP model calculates the expected wait time for each treatment job by adding up the estimated treatment times for all patients already in the queue.

On the basis of the anticipated waiting time, an HQR system is presented. Each patient should be prescribed a treatment plan that is both convenient and time-efficient.

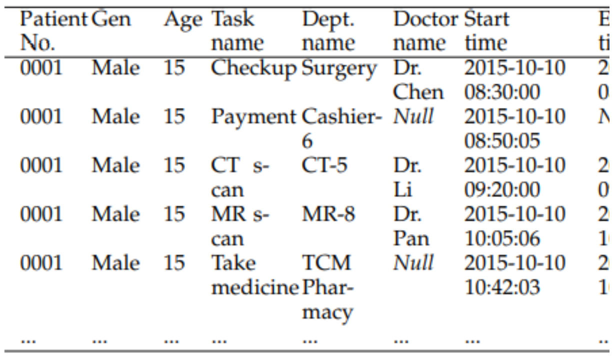
In order to accomplish the aforementioned objectives, the NSCC's PTTP and HQR systems are parallelized on the Apache Spark cloud platform. MapReduce and Resilient Distributed Datasets (RDD) programming models are used to store extensive healthcare data in Apache HBase. The rest of the paper is arranged in this way. Related studies are discussed in Section 2. The PTTP algorithm and the HQR system are described in Section 3. Section 4 explains how to run the PTTP algorithm and HQR system in parallel on the Apache Spark cloud environment. Detailed findings and assessments of the recommendation accuracy and performance are reported in Section 5. Finally, Section 6 sums up the paper's findings and suggests further research and study plans.

RELATED WORK

Classification and regression algorithms have many optimization strategies offered to increase their performance while using continuous characteristics in the data. Binary regression trees may be constructed incrementally using a self-adaptive induction approach introduced in [1]. Parallel boosted regression tree approach for web search ranking was presented by Tyree et al. [2]. A decision tree method based on a correlation-splitting criteria was developed in [3]. Classification and regression tree approaches enhanced in [4–6] have also been developed. Big data mining may benefit from the random forest method [7], an ensemble classifier based on a decision tree. Some applications of this algorithm include fast action detection via discriminative random forest voting and Top-K sub volume search[8, 9], robust and accurate shape model matching via random forest regression voting [9, 10], and a big data analysis framework for the detection of botnets among peers using random forests. The random forest algorithm's efficacy and adaptability are shown in these papers. There has been a recent push to increase the random forest algorithm's precision by Bernard [11]. Classifying high-dimensional, three-dimensional, noisy data using a random forest technique based on weighted trees was presented in [12]. Although it is a typical direct voting technique, it is still used in the original random forest algorithm. In this scenario, the testing dataset's projected value would most likely be wrong due to the random forest's noisy decision trees [13]. Algorithms for suggestion have been used in a variety of industries. MapReduce-based service recommendation for large data applications was suggested by Meng et al.[14]. In [15], a recommendation system based on people's characteristics and the sorts of travel groups they belong to was put out. For online social networks, Zu et al. [16] developed a Bayesian-inference-based recommendation system, in which a user's content rating query is propagated among his direct and indirect connections. Recommendation algorithms for multi-criteria rating systems have been developed by Adomavicius et al [17]. [18] Gediminas et al. described the current generation of recommendation algorithms, such as content-based, collaborative and hybrid methods. However, present studies do not have a methodology for accurately predicting the amount of time patients would need to spend receiving therapy. Rapidity of data mining and analysis is critical in the age of big data. Supercomputers and cloud computing provide high-speed processing capability.. One of the most well-known cloud computing systems is Apache Hadoop [20] while the other is Spark [21]. For example, the MapReduce [22] architecture has been used to develop many parallel data mining techniques. Based on the MapReduce programming architecture, a variety of data-mining methods have been presented in [24–27]. Suitable for data mining and machine learning, Apache Spark is an effective cloud platform. Data are stored in memory in the Spark, and subsequent iterations of the same data are retrieved directly from the cache. Zaharia [28] demonstrated how to use Spark to do quick and interactive analytics on Hadoop data. We utilise the random forest method to train the patient treatment time consumption based on both patient and time parameters, and then develop the PTTP model, in order to estimate the waiting time for each treatment task. The RF technique uses a Classification and Regression Tree (CART) model as a meta-classifier since patient treatment time consumption is a continuous quantity. In this study, the original RF method is enhanced in four ways to get an effective result from large-scale, high-dimensional, continuous, and noisy patient data because of its inadequacies. With the original RF method, PTTP provides considerable benefits in terms of accuracy and performance over the original RF algorithm. Furthermore, there is no previous study on queue management and advice in hospitals. Using the PTTP concept, we've devised an HQR system. When it comes to queuing service computing, we believe this study is the first effort to address the issue of patient wait times in hospitals. An efficient and convenient treatment plan with the least amount of waiting time is recommended for each patient as part of a treatment queueing recommendation. Algorithms for predicting how long a patient's treatment will take A PTTP method is developed to create the PTTP model based on both patient and temporal variables. To train the PTTP model, we use enormous, complicated, noisy hospital treatment data. 3.1 Definition of the Problem and Data Preparation 3.1.1 What is the problem? Analyzing and analysing enormous amounts of noisy patient data from numerous hospitals to make predictions is a difficult endeavour. The following are some of the most pressing issues: Data in hospitals is often large, unstructured, and high-dimensional. Daily, hospitals generate enormous amounts of business data including a wealth of information on patients and their care, including demographics, diagnoses and prognostications, medical procedures, and other specifics. There is also a large amount of incomplete or inconsistent data due to the manual operation and unexpected events during treatments, such as a lack of patient gender and age data, time inconsistencies caused by the time zone settings of medical machines from different manufacturers, and treatment records with only a start time but no finish time. Treatment tasks in each department may not be completed within a certain time frame due to several factors including task content and the time of day as well as varied patient populations. For example, a CT scan takes more time on average for an elderly person than it does for a young one. (3) Hospital queue management and suggestion are subject to stringent time constraints. The PTTP model and HQR scheme must also be executed at a rapid pace. 3.1.2 Preparation of Data Preprocessing involves gathering patient treatment data from a variety of treatment activities. Every day, hospitals see a large number of patients. Let S be a group of patients at a hospital, and a patient who has been registered and whose information is represented by si. " S = s1, s2, etc., where sN is the total number of patients in S. Each patient si has a unique set of attributes, such as a unique identifier (e.g., ID), gender, age, and location. Depending on our goals, we can make good use of some of these criteria, but not all of them. Depending on their health, each patient may go to a variety of therapy locations. This is the collection of therapeutic tasks that will be performed on a certain patient, si, at that time: It is possible to have more than one record for each treatment task, such as the name of the task (x1), the location (x2), the department (x3), start and finish times, the doctor or attending staff (XK), and so on.

where yj is a feature variable of the record of treatment task xi . Here, for a single visit, we have a single record for patient name, age, gender, and multiple records for treatment tasks, as shown in Table 1.

TABLE 1 Example of treatment records



Patient No.	Gen	Age	Task name	Dept. name	Doctor name	Start time	E ti
0001	Male	15	Checkup	Surgery	Dr. Chen	2015-10-10 08:30:00	2 0
0001	Male	15	Payment	Cashier-6	*Null*	2015-10-10 08:50:05	N
0001	Male	15	CT s-can	CT-5	Dr. Li	2015-10-10 09:20:00	2 0
0001	Male	15	MR s-can	MR-8	Dr. Pan	2015-10-10 10:05:06	2 1
0001	Male	15	Take medicine	TCM Phar-macy	*Null*	2015-10-10 10:42:03	2 1
...	...	...	...	...	...	...	..

The following phases represent the preprocessing task's workflow.Analyze the results of various treatment procedures. Depending on the statistics, a medium-sized hospital sees between 8,000 and 12,000 patients every day, and between 120,000 and 200,000 records of remedial therapy. Medical examinations, registration, prescription administration and payment are only a few examples of the many therapeutic procedures that generate this data. Table 2 shows the data formats for various treatment jobs.

PTTP Model based on the improved RF Algorithm

It is necessary to first calculate the patient treatment time consumption based on various patient features and time factors in order to anticipate the waiting time for each patient treatment activity. The amount of time it takes to complete each therapy job varies depending on the task's content, the time of year, and the state of the patient. Since the PTTP model is built using patient and time characteristics, we first train it using RF. The RF method has been enhanced in four ways to better handle large-scale, high-dimensional, continuous, and noisy hospital treatment data because of the constraints of the original RF algorithm and the characteristics of the data. As data features are restricted and superfluous information like patient name, ad dress, and phone number has already been removed, the original RF method does not employ random selection but instead uses all of the chosen (cleaned) aspects of the data in the training phase. CART models are utilised as meta-classifiers in the enhanced RF algorithm since the goal variable of treatment data is patient treatment time consumption, a continuous variable. Some of the data's independent variables, such as time range (0 - 23) and day of the week (0 - 7), are nominal data, which have varying values (Monday - Sunday). This is a situation where classic CART's two-fork tree model cannot adequately represent the analysis findings. As a result, instead of using the typical CART algorithm's two-fork model, a multi-branch model is suggested for building the regression tree model successfully. Although some of the inaccuracy has been eliminated in the preparation, there may still be other sorts of noise in the data. In certain treatment jobs, the amount of time it takes to complete one patient's treatment is measured by the time it takes to complete the next patient's treatment. Suppose that the final patient in the morning had an operation time of "12:00:00" and that the first patient in the afternoon had an operation time of "14:00:00" as an example of a payment job. As "7200 (s)" is more than the expected value of "100 (s)," this is deemed erroneous data for the former. In certain cases, as while doing a blood examination, the time consumption number "7200 (s)" has been accurate. According to the treatment data attributes, we can't just classify one value of time consumption as "noisy data." To begin with, we need to detect and eliminate any errant data. Noisy data are eliminated from the average value of the data in each regression tree leaf node before accuracy is calculated. A standard direct voting approach is used in the original RF algorithm. This is because an RF with noisy decision trees would provide a projected value for the testing dataset that is off by the expected amount. A weighted voting approach was used in this study to estimate the RF model's output, as shown in Figure 1. For the sake of voting on the test data, each tree classifier is assigned a certain reasonable weight. When a tree classifier is trained, it will have a high voting weight because of its accuracy. Overall classification accuracy is improved and generalisation error is reduced by using the classifier. Based on the enhanced RF algorithm, our PTTP algorithm provides considerable improvements in terms of accuracy and performance when compared to the original RF algorithm

Training CART Regression Trees of the RF Model

The single decision tree in the RF model is a regression tree since the goal feature variable of treatment data S, the patient treatment time consumption, is a continuous value. CART regression tree models are built for each training subset straini in this manner. CART trees are generated in the RF algorithm's initial stage of optimization. Instead of randomly selecting m features from each strain of training data, the original RF technique uses all M features from all strains of training data. CART's regression tree-building technique is outlined here.

Calculate the best splitting feature variables and the best split point

By using vp in the feature subspace yj, split the training dataset in half. There are two data subsets: RL(yj, vp) and RR(yj, vp), which represent the left and right data sets, respectively. Here are the definitions for these subsets:

$$R_{L(y_j,\ v_p)} = \{x|(y_j \le v_p)\},$$
$$R_{R(y_j,\ v_p)} = \{x|(y_j > v_p)\}.$$

Construct multi-branch for the CART model.

Some independent variables of data, such as the time range (0 - 23) and the day of the week, are nominal data, which may have varying values (Monday - Sunday). The second optimization feature of the RF technique is to utilise a multi-branch regression tree model instead of a two-fork tree model to generate the CART model. Step 2 divide the tree node into two forks by selecting the variable yj and the value vp and calculating the optimal split points vpL and vpR for the left and right branches, respectively. To illustrate, let's look at the left branch and see what the optimal split point is for the present feature subspace:

$$\Phi(v_{pL}|y_j) = \max_i \Phi(v_i|y).$$

The $\Phi(v_i|y)$ is defined as follows:

$$\Phi(v_i|y) = 2P_L P_R \sum_{j=1}^{m} |p(c_j|y_L) - p(c_j|y_R)|,$$

in where PL and PR are proportions of data in the left and right branches, respectively, to the total volume of training data. A p(cj |yL) is the volume of cj data in the left branch compared to all data in this branch. Nodes on the left branch will continue to split if the split value of (vpL|yj) is bigger than the previous node, i.e. (vpL|yj) vp|YJ). If not, the final feature variables are still being calculated. The right branch is also computed in the same manner. It is then computed sequentially for each node and its two sub-nodes. A node merger should be performed if the variable split is the same in both the parent and child nodes. As a result, a multi-branch node is formed in the tree. Fig. 3 shows a CART model example of multi-branch splitting.

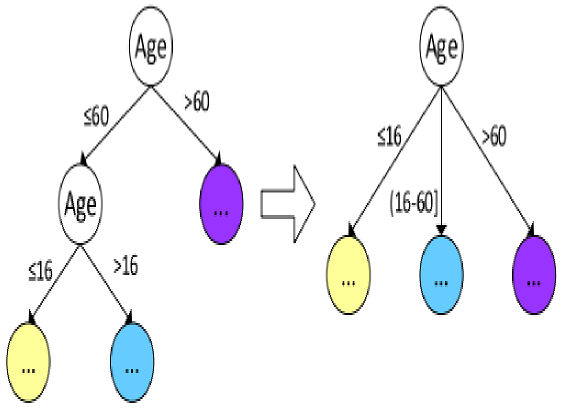


Fig. 3. Example of multi-branch splitting for the CART modeRepeat steps 1 to 3 for each branch until all of the data in that branch is categorised as a leaf node. After removing noise from the data, calculate the mean

value of leaf nodes. Although some incorrect data was eliminated in the preprocessing, there may still be other sorts of noise listed above. For this reason it is important to decrease how much of an impact noise has on RF algorithm accuracy. For each CART leaf node, a noise-removal approach based on box plots is used. Each node's data is arranged by ascending order. The box-plot model's three data points Q1, Q2, and Q3 are then computed, with Q2 being the median and Q1 and Q3 representing the data's bottom and higher four digits, respectively. An example of how to define the noise's upper limit is:

$$IL = Q1 - 1.5(IQR) = Q1 - 1.5(Q3 - Q1).$$

$$OL = Q3 + 1.5(IQR) = Q3 + 1.5(Q3 - Q1). \quad (6)$$

The data outside of the range of IL, OL" is considered to be unreliable. The average value cj of the data yj is obtained at each leaf node of the regression tree after removing the noisy data.. Formulas are defined in this way:

$$c_j = \frac{1}{k}\sum y_j, \quad (IL \le y_j \le OL),$$

3.3 Hospital Queuing Recommendation System based on PTTP Model

It is thus possible to use the PTTP model for each treatment job to construct a queue suggestion system for hospitals. To accomplish intelligent triage, a treatment plan that is both efficient and convenient is developed and suggested to each patient. For each patient, suppose there are different therapy chores based on the patient's condition, such as tests and inspections. There are a number of treatment activities that must be completed by the present patient, and there are a number of patients who are waiting in the queue for those chores to be completed. Figure 6 depicts the HQR system's workflow based on the PTTP paradigm. The present patient's treatment chores will take a certain amount of time to complete. As long as there is a patient Uik in the Task Queue,

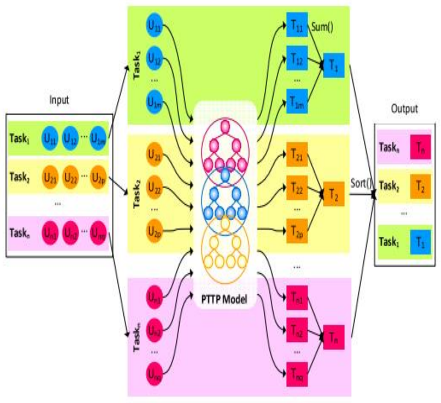


Fig. 6. Process of the HQR system based on the PTTP model

The trained PTTP model uses the patient's attributes (such as gender and age), temporal variables (such as the week and month of the present time), and other factors to forecast the patient's treatment time consumption (such as treatment departments, available machines, and service windows). Tik is the patient treatment time used by patient Uik in line.

4PARALLEL IMPLEMENTATION OF THE PTTP

HQR SYSTEM AND ALGORITHM There are almost 5 TB of past treatment data (increasing every day) saved in HBase. The PTTP model and the HQR system are then parallelized on the Apache Spark cloud platform for better performance. Thus, the algorithms' performance is greatly enhanced. This is an RDD. The training dataset is saved as an RDD object called RDDoriginal. For each RDDtraini object, k subsets of the training set are sampled from RDDoriginal. In order to store related OOB subsets, further k RDD objects are constructed, each of which is specified as RDDOOBi. K map jobs are assigned to numerous slave nodes at the same time, with k training subsets. The RDD programming paradigm and a sequence of procedures are used in parallel to compute these training subsets. In the end, there are k regression tree models Transformation and action are two sorts of operations supported by each RDD object in the RDD programming paradigm. Map(), filter(), flatM ap(), mapPartitions(), union(), and join() are examples of transformation operations on an RDD object (). It is thus possible to return a new RDD object from each change that takes place. An RDD object may be reduced, collected, counted on, saved to a Hadoop file, or counted by key in a sequence of action activities. In Algorithm 4.1, the PTTP model's dual parallelization training phases are laid out in great detail. The following are the phases in the training procedures for each RDDtraini and RDDOOBi OOB subset. A transformation and an action operation are performed in stage 1 by the buildFeatureData() and findSplitsFeature() methods. There are M feature variables referenced by M partitions in the buildFeatureData() method, which maps RDDtraini feature subspaces to a new RDD object. Each feature variable subspace's loss function and each variable's potential split point value are determined. F indSplitsFeature() is used to sort feature variables' loss function results, and then selects the lowest-valued one as Ti's initial node, which is produced as an RDD object as part of the CART tree. Two split() methods and a FindBestSplits() function are available in stage two. RDDtraini is divided into two forks by a split point in the current feature subspace (RDDL/Rtree) in the first split() function. A f indBestSplits() function is provided for each branch. When using the f indBestSplits() function, the current feature subspace's possible splitting values are determined using the same set of feature variables. RDDsplitf eature2 is found to be the optimal split point for the data in the branch. In this case, the branch continues to divide by the current feature variable and the best possible split point in the second split() method. Without further ado, let's have a look at the remaining features. To calculate the next feature, repeat steps 1 and 2 if the current node in the tree is not a leaf node. For a leaf node, move to stage 3 instead. Alternatively NoiseDataClear() and mean() functions are available in stage 3. Using the noisyDataClear() method, each leaf node's noisy data is cleaned up. This value is then averaged in order to get a value that corresponds to RDDT leaf node i. After each split, the feature variables are recalculated to complete the picture. The training subset RDDtraini is used to train a tree model RDDT i. When it comes to the accuracy of the tree RDDT I the OOB subset against the RDDtraini training subset is used for testing, and the weight in the getAccuracy() function is utilised to determine the accuracy of RDDTi. Using the cloud-computing platform to its fullest potential

CONCLUSIONS

The Apache Spark cloud environment and a large data PTTP method are introduced in this work. The PTTP model is optimised using a random forest technique. The PTTP model is used to anticipate the wait time for each treatment activity. Patients are given an efficient and easy treatment plan based on a parallel HQR system. Our PTTP algorithm and HQR system have shown outstanding accuracy and performance in several trials and real-world applications. There is an ever-increasing amount of data being generated by hospitals every day. Each set of hospital guide recommendations is anticipated to need a significant amount of time and effort to train the historical data, although this does not have to be the case. Future work should include an incremental PTTP algorithm based on real-time streaming data, as well as a more user-friendly suggestion with less path knowledge.

ACKNOWLEDGMENT

It was partially funded by the National Natural Science Foundation of China (Grant Nos. 60133005, 601432005), the National Natural Science Foundation of China (Grant Nos. 601370095 and 601472124), the International Science & Technology Cooperation Program of China (Grant No. 2015DFA11240), and the National Research Foundation of Qatar (NPRP, Grant Nos. 8-519-1-108). (Grant Nos. 2016JJ4002).

REFERENCES

[1] R. Fidalgo-Merino and M. Nunez, "Self-adaptive induction of regression trees," Pattern Analysis and Machine Intelligence, IEEE Transactions on, vol. 33, no. 8, pp. 1659–1672, 2011.

[2] S. Tyree, K. Q. Weinberger, K. Agrawal, and J. Paykin, "Parallel boosted regression trees for web search ranking," in In Proceedings of the 20th international conference on World wide web(WWW'11). ACM, 2012, pp. 387–396. [3] N. Salehi-Moghaddami, H. S. Yazdi, and H. Poostchi, "Correlation based splitting criterionin multi branch

decision tree," Central European Journal of Computer Science, vol. 1, no. 2, pp. 205–220, June 2011. [4] G. Chrysos, P. Dagritzikos, I. Papaefstathiou, and A. Dollas, "Hc-cart: A parallel system implementation of data mining classification and regression tree (cart) algorithm on a multi-fpga system," ACM Transactions on Architecture and Code Optimization, vol. 9, no. 4, pp. 47:1–25, January 2013. [5] N. Uyen and T. Chung, "A new framework for distributed boosting algorithm," in Proceeding FGCN '07 Proceedings of the Future Generation Communication and Networking. IEEE, 2007, pp. 420–423. [6] Y. Ben-Haim and E. Tom-Tov, "A streaming parallel decision tree algorithm," Journal of Machine Learning Research, vol. 11, no. 1, p. 849C872, October 2010. [7] L. Breiman, "Random forests," Machine Learning, vol. 45, no. 1, pp. 5–32, October 2001. [8] G. Yu, N. A. Goussies, J. Yuan, and Z. Liu, "Fast action detection via discriminative random forest voting and top-k subvolume search," Multimedia, IEEE Transactions on, vol. 13, no. 3, pp. 507 – 517, June 2011. [9] C. Lindner, P. A. Bromiley, M. C. Ionita, and T. F. Cootes, "Robust and accurate shape model matching using random forest regression-voting," Pattern Analysis and Machine Intelligence, IEEE Transactions on, vol. 25, no. 3, pp. 1–14, December 2014. [10] K. Singh, S. C. Guntuku, A. Thakur, and C. Hota, "Big data analytics framework for peer-to-peer botnet detection using random forests," Information Sciences, vol. 278, pp. 488–497, 2014.[11] S. Bernard, S. Adam, and L. Heutte, "Dynamic random forests," Pattern Recognition Letters, vol. 33, no. 12, pp. 1580–1586, September 2012. [12] H. B. Li, W. Wang, H. W. Ding, and J. Dong, "Trees weighting random forest method for classifying highdimensional noisy data," in IEEE International Conference on E-Business Engineering, vol. 10, November 2010, pp. 160–163. [13] G. Biau, "Analysis of a random forests model," Journal of Machine Learning Research, vol. 13, no. 1, pp. 1063 – 1095, January 2012. [14] S. Meng, W. Dou, X. Zhang, and J. Chen, "Kasr: A keyword-aware service recommendation method on mapreduce for big data applications," Parallel and Distributed Systems, IEEE Transactions on, vol. 25, no. 12, pp. 3221 – 3231, 2014. [15] Y. Y. Chen, A.-J. Cheng, and W. H. Hsu, "Travel recommendation by mining people attributes and travel group types from community-contributed photos," Multimedia, IEEE Transactions, vol. 15, no. 6, pp. 1283– 1295, 2013. [16] X. Yang, Y. Guo, and Y. Liu, "Bayesian-inference based recommendation in online social networks," Parallel and Distributed Systems, IEEE Transactions on, vol. 24, no. 4, pp. 642–651, 2013[17] G. Adomavicius and Y. Kwon, "New recommendation techniques for multicriteria rating systems," Intelligent Systems, IEEE, vol. 22, no. 3, pp. 48–55, 2007. [18] G. Adomavicius and A. Tuzhilin, "Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions," Knowledge and Data Engineering, IEEE Transactions on, vol. 17, no. 6, pp. 734–749, 2005. [19] X. Wu, X. Zhu, and G. Wu, "Data mining with big data," Knowledge and Data Engineering, IEEE Transactions on, vol. 26, no. 1, pp. 97–107, January 2014. [20] Apache, "Hadoop," Website, January 2015, http: