3rd National Conference on Enhancement in Biomedical Engineering and Healthcare
Organised by Department of Biomedical Engineering,
Adhiyamaan College of Engineering, Hosur, Tamil Nadu, India
© 2020 | International Journal of Scientific Research in Science and Technology

Early Stage Prediction of Caesarean Vs Normal Vaginal Delivery Using Artificial Intelligence

Naveena B[1], Pavithra U[2], Shalini R[3], Sivaranjini S[4], Nishanthini R[5]

[1,2,3,4]Students in Department of biomedical Engineering, Dhanalakshmi Srinivasan Institute of Technology, Trichy, Tamilnadu, India

[5]Assistant professor in Department of Biomedical Engineering, Dhanalakshmi Srinivasan Institute of Technology, Trichy, Tamilnadu, India

ABSTRACT

Machine learning techniques provide learning mechanism that can be used to induce knowledge from data. A few studies exist on the use of machine learning techniques for medical diagnosis, prediction and treatment. In this study we evaluate different machine learning techniques for birth classification (cesarean or normal). Data on cesarean section is collected and different medical factors are identified that result in cesarean births. A birth classification model is built using decision tree and artificial neural networks. In this paper, we provide method of classifying caesarean section and normal vaginal deliveries using fetal heart rate signals and uterine contractions using Artificial intelligence. Here we predict the status of fetal using machine learning technique "Decision tree Algorithm" which classifies the delivery of women. This gives the prediction results as Normal, Suspicious, Pathologic as accuracy above 95% with the given dataset. It can classify the births into normal and cesarean with an average accuracy, precision and recall of 98% respectively.

Keywords : Early Prediction of Delivery, CTG (Cardio Toco Graphy)

I. INTRODUCTION

Worldwide, over 130 million babies are born each year. 3.6 million will die due to perinatal complication and 1 million of these will be intrapartum still births . In the USA, the number of deliveries in 2017 was 3952,841; one in every 164 of these resulted in stillbirth.1 In the UK, in the same year, there were 671,255 with one in every 200 being stillbirth2 and 300 that died in the first four weeks of life . Cardiotocography (CTG) is the most common method used to monitor the fetus during the early stages of delivery and clinical decisions are made using the visual inspection of CTG traces. However, the main weakness with this approach is poor human interpretation which leads to high inter- and intra-observer variability.

Problem identification: Cardiotocography (CTG) is used to monitor the foetus during the early stages of delivery.

clinical decisions - visual inspection of CTG traces. weakness -poor human interpretation which leads to high inter- and intra-observer variability

Inter and intra-observer variability and low positive prediction is accountable for the 3.6 million babies that die each year.

II. METHODOLOGY

(A) Dataset preparation
(B) Data preprocessing
(C) Training and Testing
(D) Decision tree Algorithm
(E) Training and Testing Accuracy

In this proposed model, we are using Machine learning Technique to predict the fetal status using fetal heart rate signal and uterine contraction. Knowledge engineering and machine learning are used to extract disease patterns from the available medical data. The extracted patterns can be used for medical diagnosis, prediction and treatment. This paper has covered three goals: First, it has identified the significant factors that influence the type of birth. Second, it has presented a prediction model for the type of birth that can help doctors and patients We provide "Decision tree Classifier Algorithm" to classify fetal state based on statistical measures applicable to them, such as standard deviation and kurtosis.[5]

The values may be numbers, such as real numbers or integers, for example representing a person's height in centimeters, but may also be nominal data (i.e., not consisting of numerical values), for example representing a person's ethnicity. More generally, values may be of any of the kinds described as a level of measurement. For each variable, the values are normally all of the same kind. However, there may also be missing values, which must be indicated in some way.

In statistics, data sets usually come from actual observations obtained by sampling a statistical population, and each row corresponds to the observations on one element of that population. Data sets may further be generated by algorithms for the purpose of testing certain kinds of software. Some modern statistical analysis software such as SPSS still present their data in the classical data set fashion. If data is missing or suspicious an imputation method may be used to complete a data set.

Based on 2126 fetal cardiotocographs were automatically processed and respective diagnostic features are measured. Using this dataset we train the machine using Decision tree classifier and predicting the result with best accuracy.

(A) DATASET PREPARATION

A data set (or dataset) is a collection of data. In the case of tabular data, a data set corresponds to one or more database tables, where every column of a table represents a particular variable, and each row corresponds to a given record of the data set in question.

In the open data discipline, data set is the unit to measure the information released in a public open data repository. The European Open Data portal aggregates more than half a million data sets. In this field other definitions have been proposed, but currently there is not an official one. Some other issues (real-time data sources, non-relational data sets, etc.) increases the difficulty to reach a consensus about it.

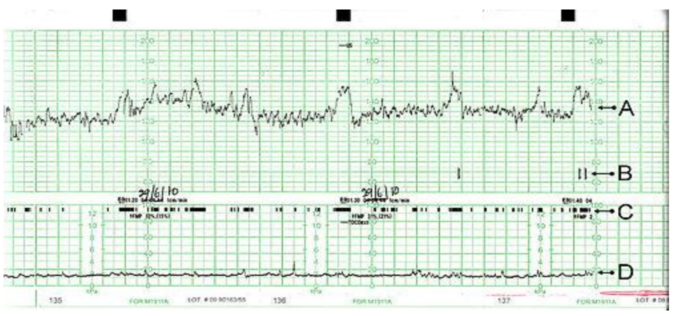
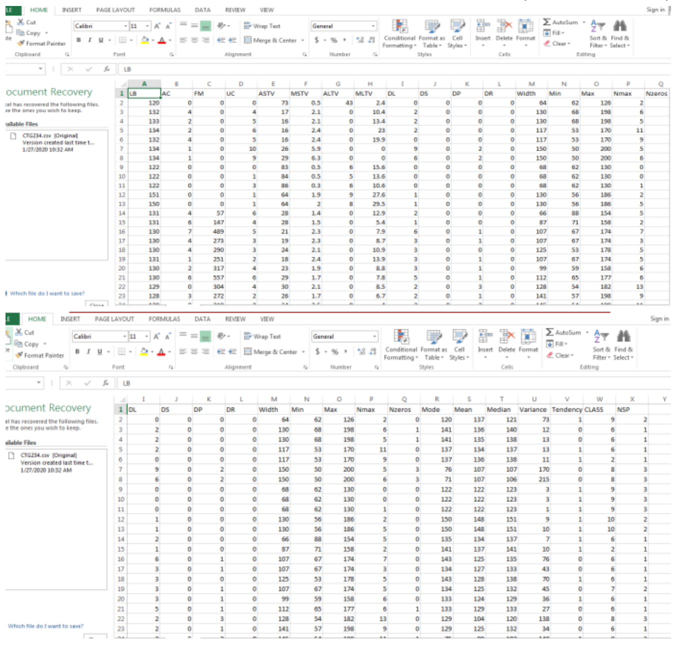


Fig.1 Cardiotocography output



	LB	AC	FM	UC	ASTV	MSTV	ALTV	MLTV	DL	DS	DP	DR	Width	Min	Max	Nmax	Nzeros
2	120	0	0	0	73	0.5	43	2.4	0	0	0	0	64	62	126	2	
3	132	4	0	4	17	2.1	0	10.4	2	0	0	0	130	68	198	6	
4	133	2	0	5	16	2.1	0	13.4	2	0	0	0	130	68	198	5	
5	134	2	0	6	16	2.4	0	23	2	0	0	0	117	53	170	11	
6	132	4	0	5	16	2.4	0	19.9	0	0	0	0	117	53	170	9	
7	134	1	0	10	26	5.9	0	0	9	0	2	0	150	50	200	5	
8	134	1	0	9	29	6.3	0	0	6	0	2	0	150	50	200	6	
9	122	0	0	0	83	0.5	6	15.6	0	0	0	0	68	62	130	0	
10	122	0	0	1	84	0.5	5	13.6	0	0	0	0	68	62	130	0	
11	122	0	0	3	86	0.3	6	10.6	0	0	0	0	68	62	130	1	
12	151	0	0	1	64	1.9	9	27.6	1	0	0	0	130	56	186	2	
13	150	0	0	1	64	2	8	29.5	1	0	0	0	130	56	186	5	
14	131	4	57	6	28	1.4	0	12.9	2	0	0	0	66	88	154	5	
15	131	6	147	4	28	1.5	0	5.4	1	0	0	0	87	71	158	2	
16	130	7	489	5	21	2.3	0	7.9	6	0	1	0	107	67	174	7	
17	130	4	273	3	19	2.3	0	8.7	3	0	1	0	107	67	174	3	
18	130	4	290	3	24	2.1	0	10.9	3	0	0	0	125	53	178	5	
19	131	1	251	2	18	2.4	0	13.9	3	0	1	0	107	67	174	5	
20	130	2	317	4	23	1.9	0	8.8	3	0	1	0	99	59	158	6	
21	130	6	557	6	29	1.7	0	7.8	5	0	1	0	112	65	177	6	
22	129	0	304	4	30	2.1	0	8.5	2	0	3	0	128	54	182	13	
23	128	3	272	2	26	1.7	0	6.7	2	0	1	0	141	57	198	9	

	DL	DS	DP	DR	Width	Min	Max	Nmax	Nzeros	Mode	Mean	Median	Variance	Tendency	CLASS	NSP
2	0	0	0	0	64	62	126	2	0	120	137	121	73	1	9	2
3	2	0	0	0	130	68	198	6	1	141	136	140	12	0	6	1
4	2	0	0	0	130	68	198	5	1	141	135	138	13	0	6	1
5	2	0	0	0	117	53	170	11	0	137	134	137	13	1	6	1
6	0	0	0	0	117	53	170	9	0	137	136	138	11	1	2	1
7	9	0	2	0	150	50	200	5	3	76	107	107	170	0	8	3
8	6	0	2	0	150	50	200	6	3	71	107	106	215	0	8	3
9	0	0	0	0	68	62	130	0	0	122	122	123	3	1	9	3
10	0	0	0	0	68	62	130	0	0	122	122	123	3	1	9	3
11	0	0	0	0	68	62	130	1	0	122	122	123	1	1	9	3
12	1	0	0	0	130	56	186	2	0	150	148	151	9	1	10	2
13	1	0	0	0	130	56	186	5	0	150	148	151	10	1	10	2
14	2	0	0	0	66	88	154	5	0	135	134	137	7	1	6	1
15	1	0	0	0	87	71	158	2	0	141	137	141	10	1	2	1
16	6	0	1	0	107	67	174	7	0	143	125	135	76	0	6	1
17	3	0	1	0	107	67	174	3	0	134	127	133	43	0	6	1
18	3	0	0	0	125	53	178	5	0	143	128	138	70	1	6	1
19	3	0	1	0	107	67	174	5	0	134	125	132	45	0	7	2
20	3	0	1	0	99	59	158	6	0	133	124	129	36	1	6	1
21	5	0	1	0	112	65	177	6	1	133	129	133	27	0	6	1
22	2	0	3	0	128	54	182	13	0	129	104	120	138	0	8	3
23	2	0	1	0	141	57	198	9	0	129	125	132	34	0	6	1

Fig.2 Dataset calculation

(B) DATA PREPROCESSING

IMPORT LIBRARIES

First step is usually importing the libraries that will be needed in the program. A library is essentially a collection of modules that can be called and used. A lot of the things in the programming world do not need to be written explicitly ever time they are required. There are functions for them, which can simply be invoked. This is a list for most popular Python libraries for Data Science. Here's a snippet of

me importing the pandas library and assigning a shortcut "pd".

Import pandas as pd

Import numpy as np

IMPORT THE DATASET

A lot of datasets come in CSV formats. We will need to locate the directory of the CSV file at first (it's more efficient to keep the dataset in the same directory as your program) and read it using a method called read_csv which can be found in the library called pandas.

import pandas as pd dataset = pd .read_csv('Medium.csv')

After inspecting our dataset carefully, we are going to create a matrix of features in our dataset (X) and

create a dependent vector (Y) with their respective observations. To read the columns, we will use iloc of pandas (used to fix the indexes for selection) which takes two parameters — [row selection, column selection].

X = dataset.iloc[:, :-1].values

as a parameter selects all. So the above piece of code selects all the rows. For columns we have :-1, which means all the columns except the last one.

TAKING CARE OF MISSING DATA IN DATASET

Sometimes you may find some data are missing in the dataset. We need to be equipped to handle the problem when we come across them. Obviously you could remove the entire line of data but what if you are unknowingly removing crucial information? Of course we would not want to do that. One of the most common idea to handle the problem is to take a mean of all the values of the same column and have it to replace the missing data.

The library that we are going to use for the task is called Scikit Learn preprocessing. It contains a class called Imputer which will help us take care of the missing data.

fromsklearn.preprocessing import Imputer

A lot of the times the next step, as you will also see later on in the article, is to create an object of the same class to call the functions that are in that class. We will call our object imputer. The Imputer class can take a few parameters —

i. missing_values — We can either give it an integer or "NaN" for it to find the missing values.
ii. strategy — we will find the average so we will set it to mean. We can also set it to median or most_frequent (for mode) as necessary.
iii. axis — we can either assign it 0 or 1, 0 to impute along columns and 1 to impute along rows.

imputer = Imputer(missing_values = "NaN", strategy = "mean", axis = 0)

Now we will fit the imputer object to our data. Fit is basically training, or in other words, imposing the model to our data.

ENCODING CATEGORICAL DATA

Sometimes our data is in qualitative form , that is we have texts as our data. We can find categories in text form. Now it gets complicated for machines to understand texts and process them, rather than numbers, since the models are based on mathematical equations and calculations. Therefore, we have to encode the categorical data.

This is an example of categorical data. In the first column, the data is in text form. We can see that there are five categories — Very, Somewhat, Not very, Not at all, Not sure — and hence the name categorical data.

So the way we do it, we will import the scikit library that we previously used. There's a class in the library called LabelEncoder which we will use for the task.

Fromsklearn. preprocessing import Label Encoder

As I have mentioned before, the next step is usually to create an object of that class. We will call our object label encoder_ X.

Label encoder _X = Label Encoder()

(C) SPLITTING THE DATASET INTO TRAINING SET AND TEST SET

Now we need to split our dataset into two sets — a Training set and a Test set. We will train our machine learning models on our training set, i.e our machine learning models will try to understand any correlations in our training set and then we will test the models on our test set to check how accurately it can predict. A general rule of the thumb is to allocate 80 % of the dataset to training set and the remaining 20% to test set. For this task, we will import test_train_split from model_selection library of scikit.

fromsklearn.model_selection import

train_test_split

Now to build our training and test sets, we will create 4 sets— X_train (training part of the matrix of features), X_test (test part of the matrix of features), Y_train (training part of the dependent variables associated with the X train sets, and therefore also the same indices) ,Y_test (test part of the dependent variables associated with the X test sets, and therefore also the same indices). We will assign to them the test_train_split, which takes the parameters — arrays (X and Y), test_size (if we give it the value 0.5, meaning 50%, it would split the dataset into half. Since an ideal choice is to allocate 20% of the dataset to test set, it is usually assigned as 0.2. 0.25 would mean 25%, just saying).

X_train, X_test, Y_train ,Y_test = train_test_split(X,Y, test_size=0.2)

Feature Scaling

The final step of data preprocessing is to apply the very important feature scaling.

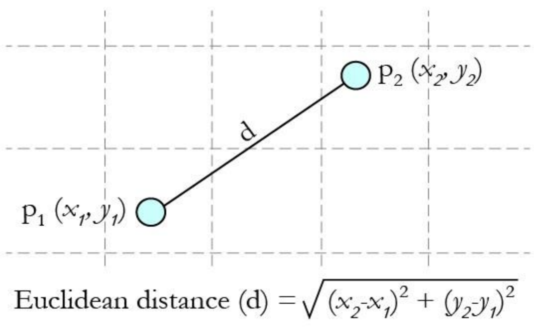


Fig.3 Feature scaling

It is a method used to standardize the range of independent variables or features of data. But why is it necessary? A lot of machine learning models are based on Euclidean distance. If, for example, the values in one column (x) is much higher than the value in another column (y), (x2-x1) squared will give a far greater value than (y2-y1) squared. So clearly, one square difference dominates over the other square difference. In the machine learning equations, the square difference with the lower value in comparison to the far greater value will almost be treated as if it does not exist. We do not want that to happen. That is why it is necessary to transform all our variables into the same scale. There are several ways of scaling the data. One way is called Standardization which may be used. For every observation of the selected column, our program will apply the formula of standardization and fit it to a s

```
fromsklearn.preprocessing import StandardScaler
sc_X = StandardScaler()
 X_train=sc_X.fit_transform(X_train)
 X_test = sc_X.transform(X_test)
```

Now we will fit and transform our X_train set (It is important to note that when applying the Standard Scalar object on our training and test sets, we can simply transform our test set but for our training set we have to at first fit it and then transform the set). That will transform all the data to a same standardized scale.

(D)DECISION TREE ALGORITHM

Decision trees are a non-parametric supervised learning method used for both classification and regression tasks. The goal is to create a model thet predicts the value of a target variable by learning simple desion rules inferred from the data features.

Decision Tree consists of:

1. **Nodes**: Test for the value of a certain attribute.
2. **Edges/ Branch**: Correspond to the outcome of a test and connect to the next node or leaf.
3. **Leaf nodes**: Terminal nodes that predict the outcome (represent class labels or class distribution).

Classification trees (Yes/No types) :

What we've seen above is an example of classification tree, where the outcome was a variable like 'fit' or 'unfit'. Here the decision variable is **Categorical/ discrete**.Such a tree is built through a process known as **binary recursive partitioning**. This is an iterative process of **splitting the data into partitions**, and then splitting it up further on each of the branches.

Creation of Decision Tree:

In this method a set of training examples is broken down into smaller and smaller subsets while at the same time an associated decision tree get incrementally developed. At the end of the learning process, a decision tree covering the training set is returned.The key idea is to use a decision tree to partition the data space into cluster (or dense) regions and empty (or sparse) regions.

In Decision Tree Classification a new example is classified by submitting it to a series of tests that determine the class label of the example. These tests are organized in a hierarchical structure called a decision tree. Decision Trees follow Divide-and-Conquer Algorithm.

Decision Tree Classifier

- Using the decision algorithm, we start at the tree root and split the data on the feature that results in the **largest information gain (IG)** (reduction in uncertainty towards the final decision).

- In an iterative process, we can then repeat this splitting procedure at each child node **until the leaves are pure**. This means that the samples at each leaf node all belong to the same class.

- In practice, we may set a **limit on the depth of the tree to prevent over fitting**. We compromise on purity here somewhat as the final leaves may still have some impurity.

Advantages of Classification with Decision Trees:

1. Inexpensive to construct.
2. Extremely fast at classifying unknown records.
3. Easy to interpret for small-sized trees
4. Accuracy comparable to other classification techniques for many simple data sets.
5. Excludes unimportant features.

(E)CONFUSION MATRIX

A confusion matrix is a technique for summarizing the performance of a classification algorithm. Classification accuracy alone can be misleading if you have an unequal number of observations in each class or if you have more than two classes in your dataset..

Calculating a confusion matrix can give you a better idea of what your classification model is getting right and what types of errors it is making.

How to calculate a confusion matrix for a 2-class classification problem from scratch.

A confusion matrix is a summary of prediction results on a classification problem.

The number of correct and incorrect predictions are summarized with count values and broken down by each class. This is the key to the confusion matrix.

The confusion matrix shows the ways in which your classification model is confused when it makes predictions. It is this breakdown that overcomes the limitation of using classification accuracy alone. our need a test dataset or a validation dataset with expected outcome values. Make a prediction for each row in your test dataset. From the expected outcomes and predictions count: The number of correct predictions for each class. The number of incorrect predictions for each class, organized by the class that was predicted.

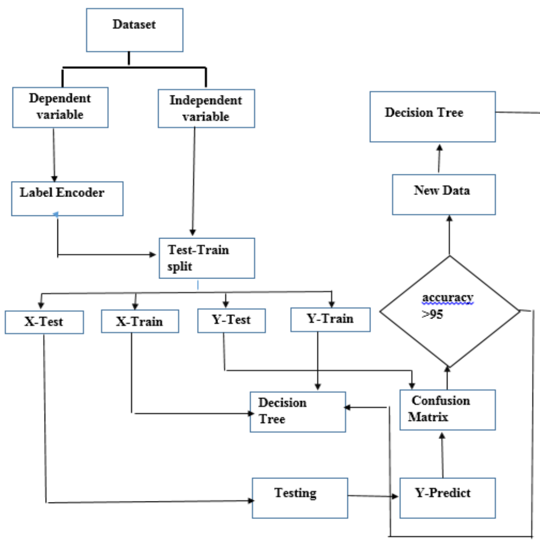
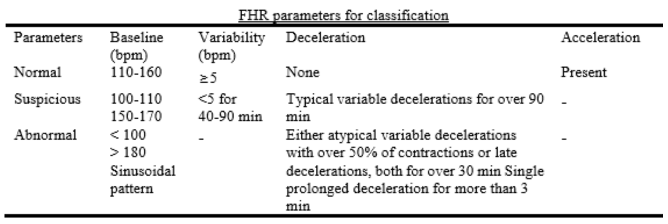


Fig.4 Flow Diagram

Parameters



FHR parameters for classification

Parameters	Baseline (bpm)	Variability (bpm)	Deceleration	Acceleration
Normal	110-160	≥5	None	Present
Suspicious	100-110 150-170	<5 for 40-90 min	Typical variable decelerations for over 90 min	-
Abnormal	< 100 > 180 Sinusoidal pattern	-	Either atypical variable decelerations with over 50% of contractions or late decelerations, both for over 30 min Single prolonged deceleration for more than 3 min	-

Fig.5 CTG

III.CONCLUSION

This paper, presented a proof-of-concept using machine learning and FHR signals and uterine contraction as an ambulatory decision support to antenatal care.

The results indicate that it is possible to provide high predictive capacity when separating normal vaginal deliveries and caesarean section deliveries and in many cases produce much better results than those reported in previous studies.

An active machine learning "Decision tree Algorithm" approach is proposed for caesarian with greater automation and better performance.

The developed approach was evaluated with actual datasets collected from the features of fetal heart rate and uterine contraction.

The evaluation process is conducted with manually labeled data and the proposed active machine learning shows a favorable performance.
The accuracy of outcome prediction is to be 98% using Decision tree algorithm . From this we can get better performance analysis

IV. FUTURE WORK

We have obtained quite satisfactory results for classification of births. However these results can be further improved by identification of additional factors that influence the type of birth. We will investigate these factors in the future. In this work three machine learning techniques have been used. We want to evaluate other techniques on the medical data. The prediction models will be more useful if available online. We will make these models online so that they can be further trained on the new data available. We want to increase the area surveyed because there may be certain geographical factors that can influence the type of birth

V. REFERENCES

[1]. Sinai Talaulikar, Vikram, and S. Arulkumaran. "Medico-Legal Issues with CTG Interpretation." Current Women S Health Reviews 2013.9(2013):145-157.

[2]. G. Georgoulas, D. Stylios and P. Groumpos, "Predicting the risk of metabolic acidosis for newborns based on fetal heart rate signal classification using support vector machines," in IEEE Transactions on Biomedical Engineering, vol. 53, no. 5, pp. 875-884, May 2006

[3]. A. Fanelli, G. Magenes, M. Campanile and M. G. Signorini, "Quantitative Assessment of Fetal Well-Being Through CTG Recordings: A New Parameter Based on Phase-Rectified Signal Average," in IEEE Journal of Biomedical and Health Informatics, vol. 17, no. 5, pp. 959-966, Sept. 2013.

[4]. V. Chudáček, J. Andén, S. Mallat, P. Abry and M. Doret, "Scattering Transform for Intrapartum Fetal Heart Rate Variability Fractal Analysis: A Case-Control Study," in IEEE Transactions on Biomedical Engineering, vol. 61, no. 4, pp. 1100-1108, April 2014.

[5]. P. A. Warrick*, E. F. Hamilton, D. Precup and R. E. Kearney, "Identification of the Dynamic Relationship Between Intrapartum Uterine Pressure and Fetal Heart Rate for Normal and Hypoxic Fetuses," in IEEE Transactions on Biomedical Engineering, vol. 56, no. 6, pp. 1587-1597, June 2009.

[6]. Goddard, Ros. "Electronic fetal monitoring." Jama 155.1(2001):1525-1526.

[7]. H. Helgason, P. Abry, P. Goncalvès, C. Gharib, P. Gaucherand and M. Doret, "Adaptive Multiscale Complexity Analysis of Fetal Heart Rate," in IEEE Transactions on Biomedical Engineering, vol. 58, no. 8, pp. 2186-2193, Aug. 2011.

[8]. S. Dash, J. G. Quirk and P. M. Djurić, "Fetal Heart Rate Classification Using Generative Models," in IEEE Transactions on Biomedical Engineering, vol. 61, no. 11, pp. 2796-2805, Nov. 2014.

[9]. G. Georgoulas, D. Stylios and P. Groumpos, "Predicting the risk of metabolic acidosis for newborns based on fetal heart rate signal classification using support vector machines," in IEEE Transactions on Biomedical Engineering, vol. 53, no. 5, pp. 875-884, May 2006.

[10]. Tsui, Sheng Yang, C. S. Liu, and C. W. Lin. "Modified maternal ECG cancellation for portable fetal heart rate monitor." Biomedical Signal Processing & Control 32(2016).

[11]. M. Lichman , UCI Machine Learning Repository, California: University of California, School of Information and Computer Science,2013.

[12]. W. Yin, X. Yang, L. Zhang and E. Oki, "ECG Monitoring System Integrated With IR-UWB Radar Based on CNN," in IEEE Access, vol. 4, pp. 6344-6351, 2016.

[13]. J. Jezewski, T. Kupka and K. Horoba, "Extraction of Fetal Heart-Rate Signal as the Time Event Series From Evenly Sampled Data Acquired Using Doppler Ultrasound Technique," in IEEE Transactions on Biomedical Engineering, vol. 55, no. 2, pp. 805-810, Feb. 2008.

[14]. Ocak, Hasan, and H. M. Ertunc. "Prediction of fetal state from the cardiotocogram recordings using adaptive neuro-fuzzy inference systems." Neural Computing & Applications 23.6(2013):1583-1589. 14 Huang, Mei Ling, and Y. Y. Hsu. "Fetal distress prediction using discriminant analysis, decision tree, and artificial neural network."Journal of Biomedical Science & Engineering 05.9(2012).

[15]. Ayresdecampos D, Spong C Y, Chandraharan E. FIGO consensus guidelines on intrapartum fetal monitoring: Cardiotocography.J. International Journal of Gynecology & Obstetrics, 2015, 131(1):13-24.

Cite this article as :

Sh