Comparison of Machine Learning Algorithms Used for Cardiotocography Classification Considering Target Labels Correlation

Ifeanyi Okwuchi, Chris Carnduff, Shan Pruthi
Systems Design Engineering, University of Waterloo, Ontario, Canada
ieokwuch/cacarndu/spruthi@uwaterloo.ca

***Abstract*—Cardiotocography (CTG) has been a widely used process to record fetal heart rate (FHR) and uterine contractions (UC) during pregnancy. The results from the CTG is analyzed and used to classify the fetus into one of several morphological patterns or fetal states. This classification has traditionally been done by obstetricians based on standard and approved guidelines but that does not eliminate the tedious nature of the task nor the high probability of classification errors. Recently, machine learning techniques have been used to make these classifications with high accuracy but no extensive comparisons to determine the best model has been done. We carry out predictions for both fetal state and morphological patterns using 7 different models and an ensemble of the best models. We also explore the correlation between the two sets of labels to see how knowledge of one of them could affect the prediction of the other. We then show that our models performed better than those of other researchers who used the UCI data set. Also, the ensemble worked better than the individual models and the correlation between the labels (fetal state and morphological pattern) improved the accuracy of predicting one label when the other one is known.**

Keywords: Machine Learning, FHR, Cardiotocography, Classification.

I. INTRODUCTION

Recent advancements in technology and health have led to drastic changes in fertility patterns around the world. According to the United Nations, over 131 million babies are born every year. Of this number, the highest fertility rates can be found in developing countries in sub-saharan Africa, such as Niger, Angola, and Mali. Despite the positive connotations surrounding childbirth, pregnancies can lead to complications, which can be life-threatening for both the mothers and their babies [16]. Approximately 303,000 women die every year from preventable causes related to pregnancy and childbirth around the world. 99% of these maternal deaths occur in developing countries, and more than half of these in sub-saharan Africa alone [15]. In addition, maternal mortality is higher amongst women living in remote locations and in poorer countries [17].

The number of maternal deaths and medical complications surrounding pregnancies continue to pose challenges today [17]. Specifically, the imbalance of maternal deaths towards developing countries portrays the lack of accessibility to medical resources and services available to pregnant women [15]. The primary reasons for this include a low number of skilled healthcare workers, poverty, and lack of information. Furthermore, according to UNICEF, more than 60% of women in developing countries don't get the recommended number of prenatal visits. As a result, the lifetime risk of maternal death is 55 times more likely in sub-saharan Africa compared to other regions of the world like Central Eastern Europe and Russia [18].

Some of the most common complications surrounding pregnancy include hypertension, sepsis, bleeding, preeclampsia, stillbirth, anemia, infections, hyperemesis gravidarum, and diabetes during pregnancy [19] [20]. Although many of these problems can be mild, more severe cases that go untreated can be life-threatening to the mother or her child. It is recommended that women receive prenatal care before and during pregnancy to help determine the health of the fetus and mitigate the level of risk associated with childbirth.

The most popular medical method for evaluating potential complications revolving around pregnancy is known as cardiotocography or CTG. This involves using ultrasound for visual analysis of fetal heartbeat and uterine contractions during pregnancy. It is widely used as a standard to assess the wellbeing of a fetus. CTG monitoring can help identify signs of hypoxia and suspicious patterns in fetal heart rates (FHR) [21]. When assessing uterine activity, key data includes frequency, duration, resting tone, interval, and intensity. For fetal heartbeats, accelerations, heart rate, periodic decelerations, and FHR variability are monitored [17] [22].

Currently, the data gathered from a fetal cardiotocograph requires extensive visual inspection before being classified by a set of expert obstetricians. The goal is to classify the health of the fetus in terms of the fetal state as well as the morphological pattern of the heart rate. Not only is this routine diagnosis redundant and resource-intensive, given the number of babies delivered each day, but it also varies from one expert to another. If there were a way to leverage core CTG inputs to classify fetal state and pattern behaviour to a high degree of confidence, this would decrease the number of resources, experts, and time needed to conduct the diagnosis. Furthermore, it would limit the variability due to subjectivity between experts, creating a consistent, automated method to classify fetal wellbeing. Finally, it would also help identify pathological or severe cases earlier on that might have been missed by obstetricians. With earlier detection, it might be possible to conduct a Caesarean section, which could potentially reduce the level of risk [2].

II. RELATED WORK

Several research papers have been written on the evaluation and classification of fetal CTGs. A wide range of classification techniques have already been implemented, including decision trees, artificial neural networks, KNNs, random forests, support vector machines, etc. The

classification problems for these techniques vary from binary classification (pathological or not), fetal state (3 class), and morphological patterns (10 class).

Stylios et al applied KNN and MLP to classify CTG signals into two classes, normal or pathological with maximum accuracies around 73% [23]. R.A.R.S Baluz used J48, Random Forests, Naive Bayes and Bagging to classify CTGs into fetal state and morphological patterns. Random Forests gave the best accuracy of 94.5% for fetal state and 87.3% for morphological pattern task [12].

Zhao et al analyzed CTGs, extracted features using statistical tests and Principal Component Analysis (PCA) then made predictions using decision trees, adaptive boosting and support vector machines and achieved accuracies of 92% and 89% for fetal state and morphological pattern classifications respectively. These accuracies were better than what was obtained from the original data set [24].

Jacob and Ramani utilized data mining techniques to analyze CTGs. They determined the impacts of outliers and showed that they helped improve classification accuracy for both the 3 and 10 class targets. They also show clusters within the data set and how each attributes contributes to the cluster generation [25].

Sundar et al applied artificial neural networks for the classification of cardiotocography data into 3 fetal states; normal, suspect and pathological. Their technique gave good results based on Rand Index, precision, recall and f1-score [4]. Miao used deep learning techniques to classify CTGs into 10 classes called morphological patterns. Their technique gave an accuracy of 88.02%, a recall of 84.30%, a precision of 85.01%, and an F-score of 0.8508 in average [17].

Jezewski et al examined the influence of feature selection methods on fetal state classification performance with Lagrangian Support Vector Machine (LSVM) based on CTG data. Three methods of feature selection were used but the best results were obtained when all features were used [26].

Kamat and Kamath applied Random Forests in classification of CTG data into 10 different morphological patterns. The best model 600 trees and 5 variables resulting in an out-of- bag (OOB) estimate of error of 12.43%. They later used a decision tree approach for the classification and it proved to have a good potential for CTG classification [27].

Wang et al collected data, calculated features, trained deep learning model, generated a new data set with four reshuffled features from the original data set and then used an ensemble classification on the new data set. The ensemble approach achieved an accuracy better than all the traditional approaches they used [28].

Sonatakke et al solved both the fetal state and morphological patterns classification problems using SVM, GLM, ANN, KNN and Random Forests. Of all their models, the best results were gotten from random forest with accuracies of 86.8% for morphological pattern and 93.4 for fetal state classification.

The various researchers who have worked on this problem have utilized different data sets. In order to make fair comparison, we will be comparing our models with those of researchers who also utilized the UCI cardiotocography data set. Some researchers solved the binary class problem, some solved the 3 class problem, some solved the 10 class problem and others solved both 3 and 10 class problems. Some used 1 or 2 techniques and others used up to 5.

Our contributions to this include solving the problem extensively with 8 different techniques on both the fetal state (3 class) problems and the morphological pattern (10 class) problems. We also go further to explore and exploit the correlation between the 3 class and 10 class labels. We experiment to find how much improvement in accuracy can be achieved in predicting morphological patterns assuming the fetal state is known and added to the data attributes. We then experiment to find how much increase in accuracy can be gotten when predicting fetal state assuming the morphological pattern is known. The reason for exploring the effect of this correlation is because currently, FIGO standards are often being used to classify CTG signals into 3 fetal states but such standards do not exist for classification into morphological patterns. This implies that it is harder for humans to classify CTGs into morphological patterns hence we are proposing a way to utilize the human made fetal state classification to aid the machine make more accurate morphological pattern classification. We also create an ensemble of the 3 best algorithms and using this ensemble, we explore if any improvements in prediction can be achieved. The algorithms we'll be using to solve our problem are listed below. We use algorithms that are representative of the various classes such as Neighbors, Support Vector Machines, Bayesian, Regression, Artificial Neural Networks, Decision Trees and Ensemble.

III. INTRODUCTION TO APPROACHES

A. Support Vector Machines (SVM)

Support vector machines introduced by Vladimir Vapnik find a decision boundary to separate classes and maximizes the margin between classes. The margin is the distance between this decision boundary and the data points, also known as support vectors, closest to the boundary [29]. In linearly separable cases, SVMs ensure that all data points are on the correct side of the decision boundary, and then aim to maximize the margin in order to find the optimal line [31]. In non-linearly separable cases, SVMs often employ the kernel trick to find a non-linear decision boundary or use a soft margin, where a few misclassified points are tolerated [30].

B. Naïve Bayes Classifier (NB)

The Naïve Bayes classifier is a simple probabilistic classifier that performs well in real world applications compared to more complex classifiers. It makes the core assumption that each features are conditionally independent, given class [32]. Then, it uses conditional and class probabilities to classify unseen data. Despite its simplicity, Naive Bayes classifier can be extremely fast, accurate and reliable for multi-class problems.

C. Gradient Boosting Machine (GBM)

This is a machine learning technique for regression and classification problems, which produces a prediction model in the form of an ensemble of weak prediction models, typically decision trees. It builds the model in a stage-wise fashion like other boosting methods do, and it generalizes them by allowing optimization of an arbitrary differentiable loss function.

The idea of gradient boosting originated in the observation by Leo Breiman that boosting can be interpreted as an optimization algorithm on a suitable cost function [35] and

further developed by J.H Friedman [36]. Gradient Boosting comprises 3 basic elements, a loss function to be optimized, a weak learner to make predictions and an additive model which adds weak learners to minimize the function.

D. Random Forest (RF)

The random forest method is a classification algorithm that uses a decision tree architecture to perform classification. After splitting the data into training and testing subsets, a number of unique trees are generated by selecting k random features from m features from a random subset of the training data. This subset of the training data is independently selected which can lead to some of the same data being present in the creation of more than one individual tree [33]. For each tree, the root node is calculated using the best split point using the Gini index. The Gini index is a measure of prediction power among a number of different variables to rank the importance of features. The next best split point is then calculated to produce a daughter node. New nodes are created until the tree reaches the terminal node to make the classification. This procedure is performed to create N trees which make up the forest [33].

To classify new inputs, the data is passed through each tree in the forest which each allot one vote to a class which results in a prediction of the most votes [34]. The testing subset is used to perform out-of-bag error estimation on the forest and to determine the models accuracy [34].

E. Logistic Regression (LR)

The multinomial logistic regression method creates a set of independent binary regressions for each class. A logistic function is used to model the probability of new data either belonging to that specific class or not. After training logistic functions for each binary regression, the model can perform a prediction by passing new data through each logistic function. The function that results in the highest probability of it belonging to a specific class is chosen as the correct classification. Multinomial logistic regression does not assume linearity, normality or homoscedasticity in the data making it an appealing method for this problem [38].

F. K-Nearest Neighbors (KNN)

The KNN method is a non parametric classification algorithm that assigns unclassified data to categories based on their Euclidean distance to classified data in hyperspace. The algorithm requires a user input of k to indicate the number of nearest neighbors that will be used to make the classification. New data is classified based on its Euclidean distance to these nearest neighbors in a majority rules fashion. This algorithm is only useful for small data sets as it must perform many distance calculations in order to make a classification [37].

G. Multi-Layer Perceptron (MLP)

A multi-layer perceptron functions similar to neurons the human brain. It contains a number of layers of two-state processors known as nodes. These nodes take in multiple inputs to process a binary output (either one or zero). The MLP consists of an input layer, 1 or more hidden layers and an output layer. Each node of the input layer is connected to the hidden layer(s) and the last hidden layer is connected to the output layer. These the connections between each node are weighted based on their individual connectivity level. This network of nodes is trained by updating these weights so that the output layer correctly classifies the input [39].

IV. DESCRIPTION OF THE DATA

The data set for this problem is obtained from the University of California Irvine online repository. This data is a cardiotocography data set which was created by researchers Ayres-de-Campos, Bernades and Marques-de-Sa in Portugal [1]. In the dataset, we have 2126 instances of the 21 unique cardiotocographic attributes with no missing values. The measurements in the attributes were classified by 3 expert obstetricians and labels were assigned to the instances. Two classification schemes were proposed. One was based on 10 possible morphological patterns and the other was based on 3 possible fetal states proposed by the International Federation of Gynaecologists and Obstetrics (FIGO) [9]. The labels in these two classification schemes are described below.

TABLE I. DESCRIPTION AND NUMBER OF INSTANCES IN OF THE MORPHOLOGICAL PATTERNS CLASSIFICATION SCHEME

Class Number	Code	Description	Number of Instances
1	A	Calm sleep	384
2	B	REM sleep	579
3	C	Calm vigilance	53
4	D	Active vigilance	81
5	SH	Shift pattern	72
6	AD	Accelerative/decelerative pattern (stress situation)	332
7	DE	Decelerative pattern (vagal stimulation)	252
8	LD	Largely declarative pattern	107
9	FS	Flat-sinusoidal pattern (pathological state)	69
10	SUSP	Suspect pattern	197

TABLE II. DESCRIPTION AND NUMBER OF INSTANCES IN OF THE FETAL STATE CLASSIFICATION SCHEME

Class Number	Code	Description	Number of Instances
1	N	Normal	1655
2	S	Suspect	295
3	P	Pathological	176

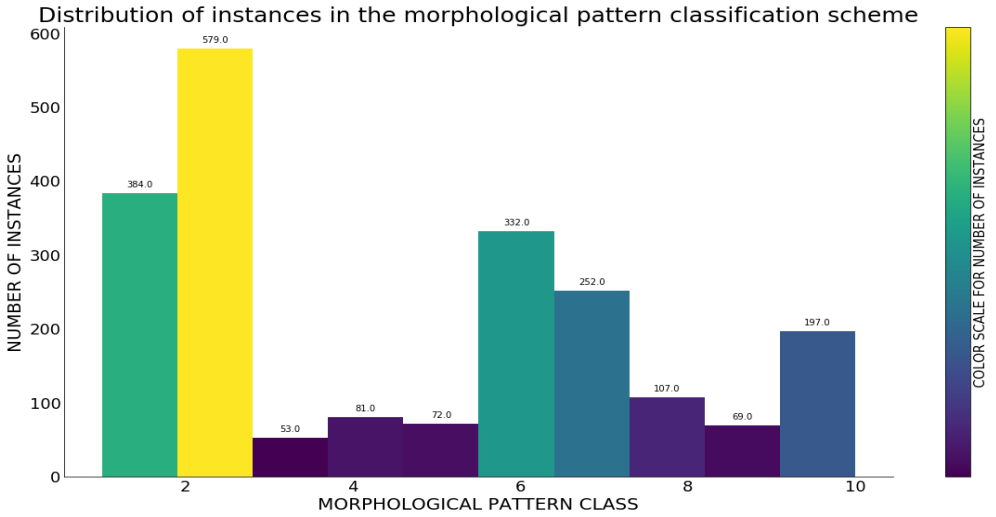

Fig. 1. Distribution of the instances in the 3 classes of the morphological pattern classification scheme.

Figures 1 and 2 show that the data set is not balanced. In the morphological pattern (10 class) classification scheme, classes 1, 2 and 6 contain 60.9% of all the instances while the remaining 7 classes share 39.1%. In the fetal state (3 class) classification scheme, class 1 (Normal) contains 77.8% of the instances while classes 2 and 3 share the remaining 22.2% of the instances.

The data set is highly unbalanced with some classes having considerably more instances than others in both classification schemes. It consists of 21 attributes which were obtained by Ayres-de-Campos et al using a SisPorto machine [1]. The attributes are real valued.

Some features in the original data set were duplicates and some had the same value for all the instances. These attributes add no value to data so they were eliminated during the pre-processing. Considering the resulting data set contains just 21 features, there was no need for any form of dimensionality reduction.

The data set can be used to carry 10 class, 3 class or binary classification experiments. Researchers carried out 2 class experiments on this data set by eliminating the suspect instances from the 3 class problem hence making it a binary classification with the targets being either normal or pathological [2] [3] [4] [5]. In this research, we would be carrying out both 10 class and 3 class experiments.

The two output classification schemes based on morphological pattern or fetal state are plotted to see if correlations exist between both. As shown in figure 3, there appears to be a positive correlation between the morphological pattern and the fetal state where a high value of morphological pattern corresponds to a high value of fetal state.

Asides just carrying out 10 class and 3 class experiments, we would also carry out experiments to see how knowing the actual values of either the morphological pattern or fetal state affects the predicting the values of the other one. Since these two classes show some correlation, the expectation is that

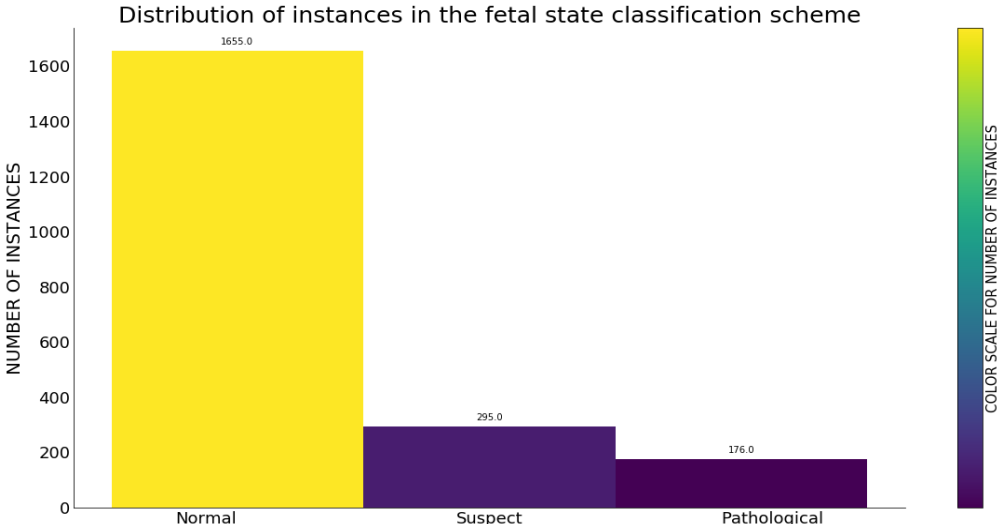

Fig. 2. Distribution of the instances in the 3 classes of the fetal state classification scheme.

having one of them as part of the features increases the ability to predict the other.

TABLE III. DESCRIPTION OF DATA ATTRIBUTES

Code	Description
LB	Baseline value
AC	Accelerations
FM	Fetal movement
UC	Uterine contractions
ASTV	% of time with abnormal short-term variability
mSTV	Mean value of short-term variability
ALTV	% of time with abnormal long-term variability
mLTV	Mean value of long-term variability
DL	Light decelerations
DS	Severe decelerations
DP	Prolonged decelerations
Width	Histogram width
Min	Low frequency of the histogram
Max	High frequency of the histogram
Nmax	Number of histogram peaks
Nzeros	Number of histogram zeros
Mode	Histogram mode
Mean	Histogram mean
Median	Histogram median
Variance	Histogram variance
Tendency	Histogram tendency

V. METHODOLOGY

The data was separated into 4 sets of features and labels corresponding to the kind of experiment to be carried out. The different sets are explained below.

A. *Set 1:*

In this set, the features used for prediction are the exact features (21 features) shown in table 3. The label being predicted here is the morphological pattern of foetus. This gives a 10 -class classification problem.

B. *Set 2:*

Here, the features used for prediction are the same as set 1 but this time the target label being predicted is the fetal state. This presents a 3-class classification problem.

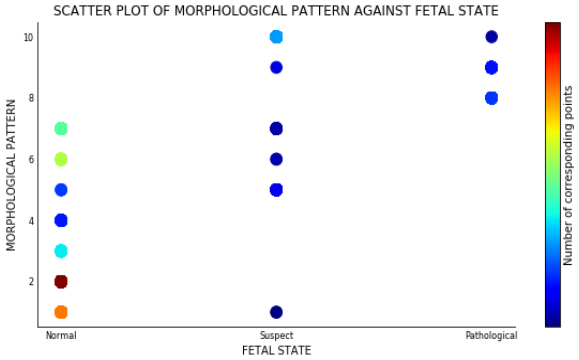

Fig. 3. Scatter plot showing the correlation between morphological pattern and fetal state for all 2126 instances in the data set with colors to show the number of instances that fall into each point.

C. *Set 3:*

For this category, we'll still be using all the attributes on table 3 but the morphological pattern (10 class labels) have been added to the features making it 22 features. With these features, we'll be predicting fetal state (3-class classification) to see how much improvement in accuracy can be derived assuming we already know the classification based on morphological pattern.

D. *Set 4:*

Finally, the features in table 3 in addition to the fetal state (3-class labels) will be used as the attributes to predict the morphological pattern. The aim here is to see how much better we can classify the morphological states (10-class) utilizing knowledge of the fetal state (3-class).

For each of the sets, the data set of 2126 instances were shuffled and split into training and test sets with a ratio of 80:20. The same splitting algorithm and random number generator was used for the four sets to ensure consistency. The data was also scaled so that algorithms such as KNN which rely heavily on distance metrics like Euclidean distance do not become biased towards the higher valued attributes.

Cross validation is a model validation method that is usually used to estimate the accuracy of a predictive model. The idea behind cross validation is that instead of using the entire dataset to train a learner, some of the data is removed prior to training, and introduced for the first time after the learner is trained [7].

5 fold cross validation was done for every algorithm as well as a parameter grid search to determine the best parameters for the model. This cross validation works by splitting the training data into 5 equal parts. One part is kept for validation while the remaining four are used for training. After that, another part is used for validation and the rest for training. At the end, there would be 5 training and validation sessions. This gives an idea as to how well the model generalizes on "new" data and the model with the best mean validation score is selected as the best [6] [7].

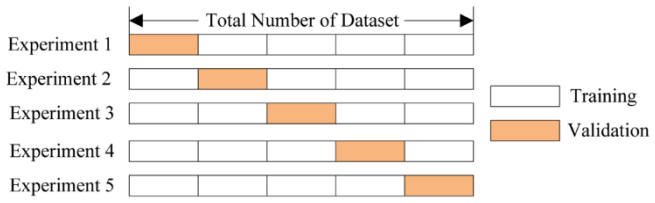

Fig. 4. 5 fold cross validation experiments

VI. RESULTS

Several highly sophisticated machine learning algorithms were tried from the various classes such as Nearest Neighbors, Ensemble, Bayesian, Artificial Neural Networks and Support Vectors. The parameters, mean cross validation accuracy, precision, recall, f1-score and test accuracy for each of these algorithms was calculated. Below are the details of these. All parameters not listed take on Python Scikit Learn default values.

A. Support Vector Machines (SVM)

A support vector classifier was built for each of the 4 sets of data. A parameter grid search was done to evaluate the best parameters (C, gamma and kernel) using a 5 fold cross validation as described in the methodology section. The best parameters were found and the model was used to predict the classes of the test data.

Our SVM model as shown in Table 4 performs significantly well in predicting both the morphological pattern and fetal state. The accuracy values reported are better than those of Sontakke et al. [8] and Chamidah [13] for both the 3-class and 10-class experiments. It is also noted that sets 3 and 4 which are the experiments where the label not being predicted was added to the data as a feature gave considerably higher accuracies. The 10 class prediction improved by 5.1% while the 3 class prediction improved by 3.3%.

B. Naïve Bayes Classifier

A Gaussian Naïve Bayes classifier was built to predict the classes for each of the four sets of data. This classifier has little or no parameters to be tuned so the default values were utilized. A 5-fold cross validation was done to see if the results generalize on new data.

Naïve Bayes classifier in Table 5 showed poor results in the classifications especially for the 10-class problems (sets 1 and 4). This might possibly be due to the naïve assumption of the model. Similarly, the poor performance on the 10 class problems could be attributed to the large number of target classes hence smaller differences in probability values associated to each class which could lead to misclassification. This explains why none of the previous works on FHR classification utilized Naïve Bayes as a classifier. We also noted that as with SVM, the sets 3 and 4 where the label not being predicted was added as a feature gave considerably higher accuracies. The 10 class prediction improved by 10.05% while the 3 class prediction improved by 6.6%.

C. Gradient Boosting Machine

The next model is a Gradient Boosting Classifier with a learning rate of 0.2. This algorithm was built for each of the four datasets. A parameter grid search was done to evaluate the best combination of parameters (number of estimators, sub sample, max depth, min samples split and min samples leaf) using a 5 fold cross validation. The best parameters were found and the model was used to predict the classes of the test data.

The results in Table 6 produced by Gradient Boosting were highly accurate. These results proved significantly better than results achieved in previous works [8] [10]. Considering that our work is the first to apply Gradient Boosting to this problem, the superiority of this algorithm in classifying CTG patterns shows promise. As with the other models, predictions for sets 3 and 4 improved which implies that knowing one of the labels helps predict the other one more accurately. The 10-class prediction improved by 4.5% (set 1 vs set 4) while the 3-class prediction improved by 3.8% (set 2 vs set 3).

D. Random Forest

Random Forest classifier was built for the 4 sets of data. An extensive parameter grid search was done to find the best parameters (number of estimators, max features, max depth, and min samples split) for each of the four sets. The best parameters are chosen from the outcome of the 5 fold cross validation and they were utilized to predict the test labels.

Random Forest model in Table 7 developed from tuning the hyper-parameters gave good results. The result beat that achieved by Kamath and Kamat [10], Sontakke et al [8], Thomas [11] and Baluz R.A.R.S [12]. The predictions for sets 3 and 4 increased compared to sets 1 and 2. This is consistent with the results from other algorithms. The 10-class prediction accuracy improved by 3.1% while the 3-class prediction improved by 4%.

E. Logistic Regression

Another method which was attempted on the four sets of data is the Logistic Regression. Parameters such as solver, penalty and multi-class were tuned using a parameter grid with a 5 fold cross validation.

As shown in Table 8, the performance of this algorithm is good. This is one algorithm that has not been previously utilized by researchers in predicting morphological patterns or fetal state from CTG. It shows consistency with the others as the accuracy of predicting sets 3 and 4 improve due to the addition of labels to the features. The 10-class predictions improved by 8% while the 3-class predictions improved by 6.1%.

F. K-Nearest Neighbors

K-Nearest Neighbors algorithms are used to classify the four sets of the data. Again, the hyper-parameters are selected by performing a grid search and a cross validation of every combination of parameters (number of neighbors, weights, p and distance metric) specified by the grid.

The K-Nearest Neighbors algorithm in Table 9 performs significantly better in the 3-class predictions compared to the 10-class prediction. Even at that, the results we obtained were significantly better than those achieved by other researchers [8]. Once again, adding one of the labels to the features improves the accuracy of predicting the other label. The 10-class predictions improved by 4.9% while the 3-class predictions improved by 3.5%.

G. Multi-Layer Perceptron

For this algorithm, we utilize an MLP with 4 hidden layers and each containing 30 neurons. Adam optimizer, random

TABLE IV. BEST RESULTS OF SVM CLASSIFICATION ON ALL 4 EXPERIMENTATION

Set	Parameters	Mean CV accuracy	Avg. Precision	Avg. Recall	Avg. F1-score	Test Accuracy
1	C=200, Gamma=0.01, Kernel=rbf	0.852	0.86	0.86	0.86	0.857
2	C=30, Gamma=0.05, Kernel=rbf	0.922	0.92	0.92	0.92	0.925
3	C=125, Gamma=0.015, Kernel=rbf	0.968	0.96	0.96	0.96	0.958
4	C=150, Gamma=0.01, Kernel=rbf	0.906	0.91	0.91	0.91	0.908

TABLE V. BEST RESULTS OF GAUSSIAN NAÏVE BAYES CLASSIFICATION ON ALL 4 EXPERIMENTATION

Set	Parameters	Mean CV accuracy	Avg. Precision	Avg. Recall	Avg. F1-score	Test Accuracy
1	N/A	0.608	0.68	0.61	0.63	0.613
2	N/A	0.808	0.86	0.78	0.81	0.784
3	N/A	0.861	0.88	0.82	0.84	0.822
4	N/A	0.706	0.80	0.72	0.74	0.718

TABLE VI. BEST RESULTS OF GRADIENT BOOSTING CLASSIFICATION ON ALL 4 EXPERIMENTATION

Set	Parameters	Mean CV accuracy	Avg. Precision	Avg. Recall	Avg. F1-score	Test Accuracy
1	n_estimators=82, subsample=1, max_depth=3, min_samples_split=3, min_samples_leaf=3	0.902	0.88	0.88	0.87	0.878
2	n_estimators=82, subsample=0.9, max_depth=3, min_samples_split=3, min_samples_leaf=3	0.960	0.95	0.95	0.95	0.948
3	n_estimators=81, subsample=1, max_depth=2, min_samples_split=4, min_samples_leaf=1	0.988	0.99	0.99	0.99	0.986
4	n_estimators=82, subsample=1, max_depth=3, min_samples_split=3, min_samples_leaf=3	0.929	0.92	0.92	0.92	0.923

TABLE VII. BEST RESULTS OF RANDOM FOREST CLASSIFICATION ON ALL 4 EXPERIMENTATION SETS

Set	Parameters	Mean CV accuracy	Avg. Precision	Avg. Recall	Avg. F1-score	Test Accuracy
1	n_estimators=800, max_features=sqrt, max_depth=90, min_samples_split=5,	0.901	0.89	0.89	0.89	0.887
2	n_estimators=400, max_features=auto, max_depth=90, min_samples_split=5	0.955	0.93	0.93	0.93	0.934
3	n_estimators=600, max_features=sqrt, max_depth=60, min_samples_split=5	0.984	0.97	0.97	0.97	0.974
4	n_estimators=1520, max_features=sqrt, max_depth=22, min_samples_split=2	0.926	0.92	0.92	0.92	0.918

TABLE VIII. BEST RESULTS OF LOGISTIC REGRESSION CLASSIFICATION ON ALL 4 EXPERIMENTATION

Set	Parameters	Mean CV accuracy	Avg. Precision	Avg. Recall	Avg. F1-score	Test Accuracy
1	solver=newton_cg, penalty=L2, multi-class=multinomial	0.831	0.84	0.83	0.83	0.831
2	solver=newton_cg, penalty=L2, multi-class=multinomial	0.893	0.88	0.88	0.88	0.878
3	solver=newton_cg, penalty=L2, multi-class=ovr	0.956	0.94	0.94	0.94	0.939
4	solver=newton_cg, penalty=L2, multi-class=multinomial	0.905	0.91	0.91	0.91	0.911

TABLE IX. BEST RESULTS OF K-NEAREST NEIGHBORS CLASSIFICATION ON ALL 4 EXPERIMENTATION SETS

Set	Parameters	Mean CV accuracy	Avg. Precision	Avg. Recall	Avg. F1-score	Test Accuracy
1	N_neighbors=3, weights=distance, p=1, metric=manhattan	0.791	0.80	0.79	0.79	0.791
2	N_neighbors=3, weights=distance, p=1, metric=manhattan	0.921	0.91	0.91	0.91	0.911
3	N_neighbors=3, weights=distance, p=1, metric=manhattan	0.962	0.94	0.95	0.94	0.946
4	N_neighbors=5, weights=distance, p=1, metric=manhattan	0.842	0.85	0.84	0.84	0.840

TABLE X. BEST RESULTS OF MULTI-LAYER PERCEPTRON CLASSIFICATION ON ALL 4 EXPERIMENTATION SETS

Set	Parameters	Mean CV accuracy	Avg. Precision	Avg. Recall	Avg. F1-score	Test Accuracy
1	Activation=relu, optimizer=adam, random state=22, batch size=8, hidden layers = 4, neurons per layer=30	0.846	0.85	0.85	0.85	0.847
2	Activation=relu, optimizer=adam, random state=22, batch size=8, hidden layers = 4, neurons per layer=30	0.925	0.91	0.91	0.91	0.913
3	Activation=relu, optimizer=adam, random state=22, batch size=8, hidden layers = 4, neurons per layer=30	0.965	0.96	0.96	0.96	0.960
4	Activation=relu, optimizer=adam, random state=22, batch size=8, hidden layers = 4, neurons per layer=30	0.890	0.91	0.91	0.91	0.908

TABLE XI. RESULTS OF ENSEMBLE CLASSIFICATION ON ALL 4 EXPERIMENTATION SETS

Set	Mean Cross Validation Accuracy	Test Accuracy
1	0.904	0.900
2	0.960	0.948
3	0.989	0.986
4	0.930	0.923

state of 22 and a Relu activation function were found to produce the best results on all four sets after extensive experimentation and trials.

The MLP as shown in Table 10 produces high accuracy scores for the classification problems. The results obtained were better than those recorded in [8] but less than that of [14]. As always, the prediction accuracy for sets 3 and 4 showed improvements due to the extra feature added to the data. Morphological pattern classification improves by 6.1% while fetal state classification improved by 4.7%.

H. Comparing the different algotithms

In each of the four sets we are experimenting with, we compare the cross validation accuracies of the several machine learning algorithms. As shown in figure 5, when predicting the morphological patterns in set 1, among the individual algorithms, Gradient Boosting (GBM) gave the best results. The mean, median and maximum acquired from gradient boosting exceeds those of all other techniques. Also gradient boosting gives a very small range and no outliers which implies that the results are consistent across all the splits. The next best model is the Random Forest which closely follows the GBM. For this set, Naive Bayes gives the worst performance compared to every other algorithm. When we consider the ensemble, we realize that it gives a better mean and median validation accuracy than all the individual models. Table 11 shows that the ensemble gives a 90% test accuracy for set 1. The closest to this is 88% achieved by random forest. This is proof that using an ensemble can actually improve the test accuracy when predicting morphological patterns.

For set 2 shown in figure 6, considering only the individual algorithms, Gradient Boosting still gives the best mean, median and maximum cross validation values. Again the range for the gradient boosting is small without any outliers making it a consistent model.

Random Forests comes in second place with very close values compared to Gradient Boosting. Naive Bayes still gives the worst results among the different classifiers but it performs significantly better on the 3 class problems compared to the 10 class problems. Now when we consider the ensemble, we realize that the mean cross validation accuracy and the test accuracy equal those of Gradient Boosting. There is a bigger range and standard deviation however in the ensemble values compared to Gradient Boosting.

The accuracy of the individual models in predicting set 3 as shown in figure 6 depicts high values for Gradient Boosting and Random Forests. Gradient Boosting comes out on top with better mean, median, minimum and maximum values. Once again, the results are compact for Gradient Boosting with a small range indicating a good consistency and generalization. Naive Bayes still remains worst model out of all of them. When we consider the ensemble, again the mean validation and test accuracies obtained equal those of the gradient boosting machine. However, ensemble values are more compact with less standard deviation than gradient boosting.

Finally, set 4 validation accuracies as given in figure 8 show that once again, for the individual models, Gradient Boosting gives the best result having the highest mean, median, maximum and minimum values. Random Forest comes close

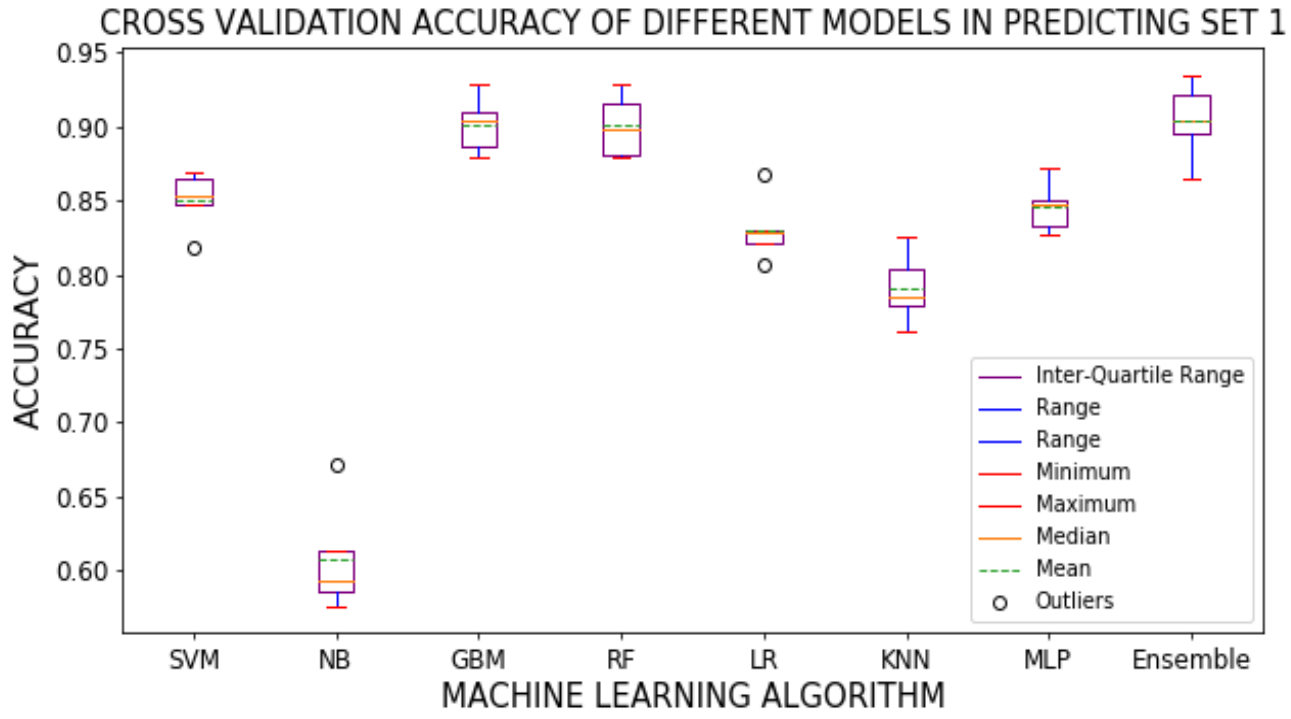

Fig. 5. Cross validation accuracy of the different machine learning algorithms on set 1 represented with a box plot. The ensemble gives the overall best result closely followed by Gradient Boosting and Random Forest. Naïve Bayes gives the worst performance.

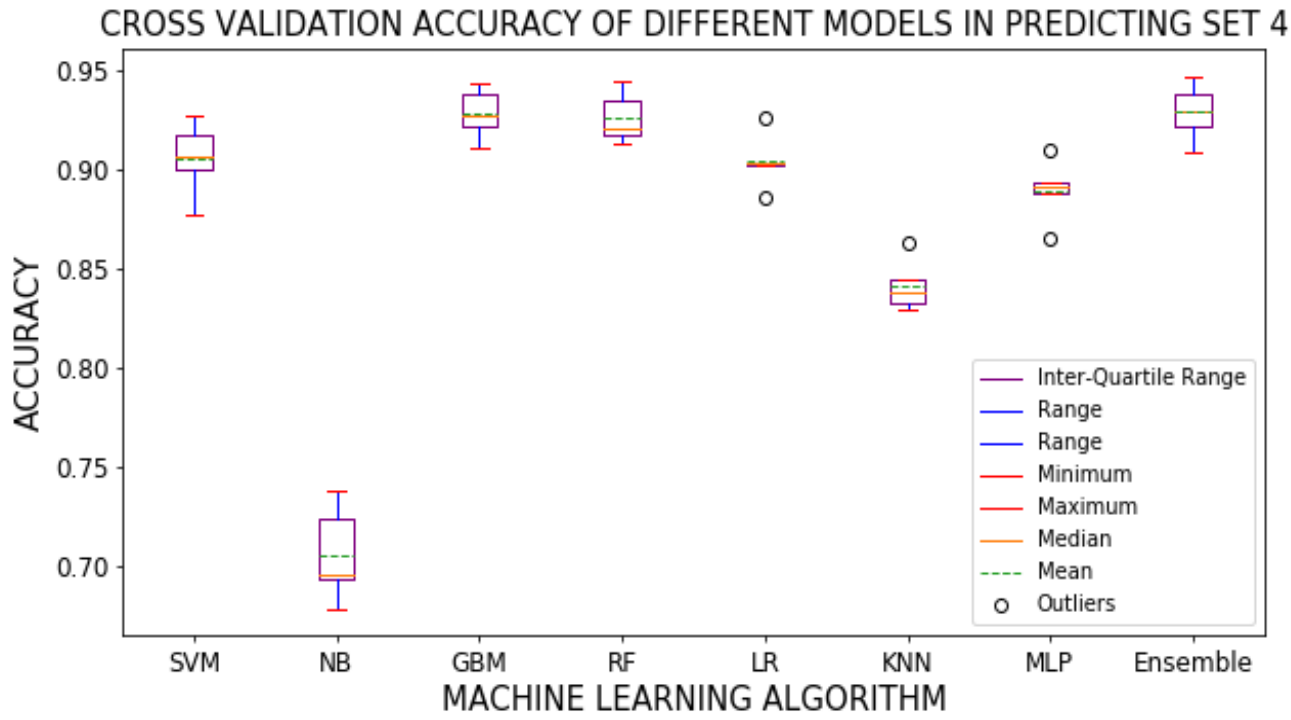

Fig. 6. Cross validation accuracy of the different machine learning algorithms on set 2 represented with a box plot. The mean cross validation scores of the ensemble and that of the GBM are equal. They both give the best performance while Naïve Bayes gives the worst performance.

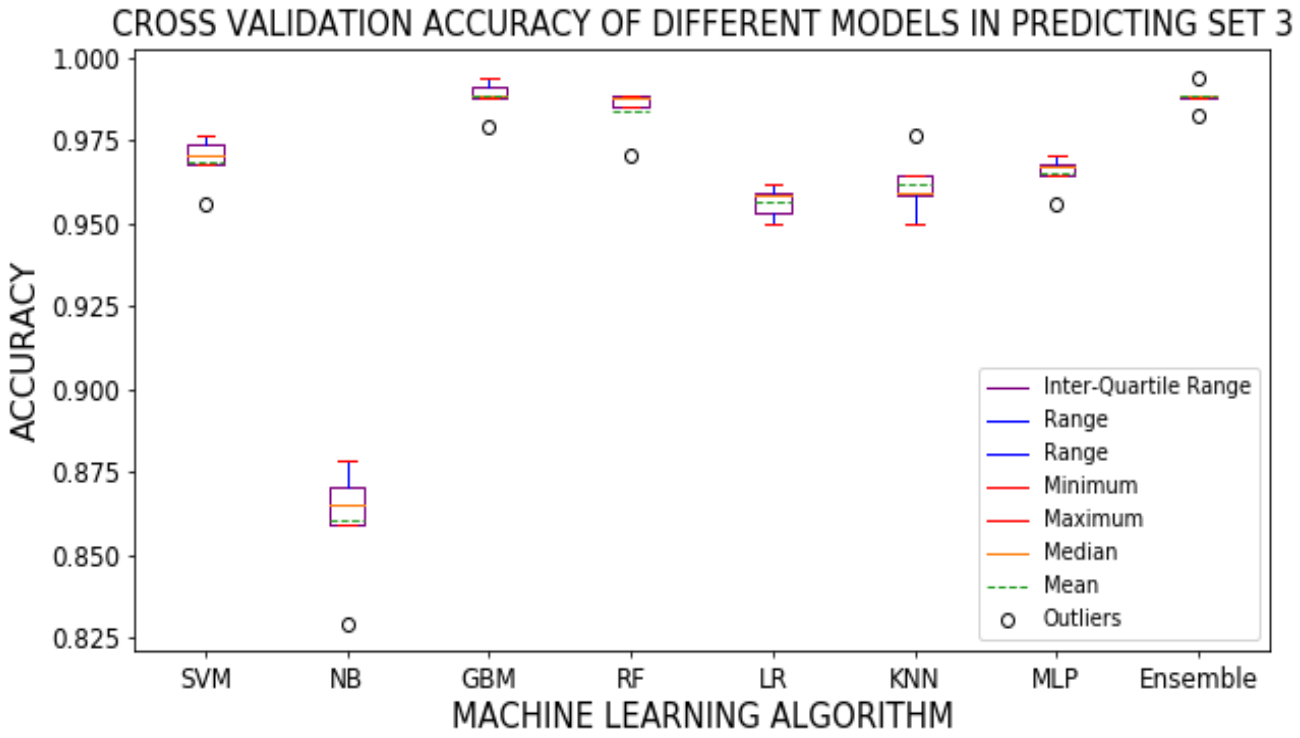

Fig. 7. Cross validation accuracy of the different machine learning algorithms on set 3 represented with a box plot. The mean cross validation scores of the ensemble and that of the GBM are equal. They both give the best performance while Naïve Bayes gives the worst performance.

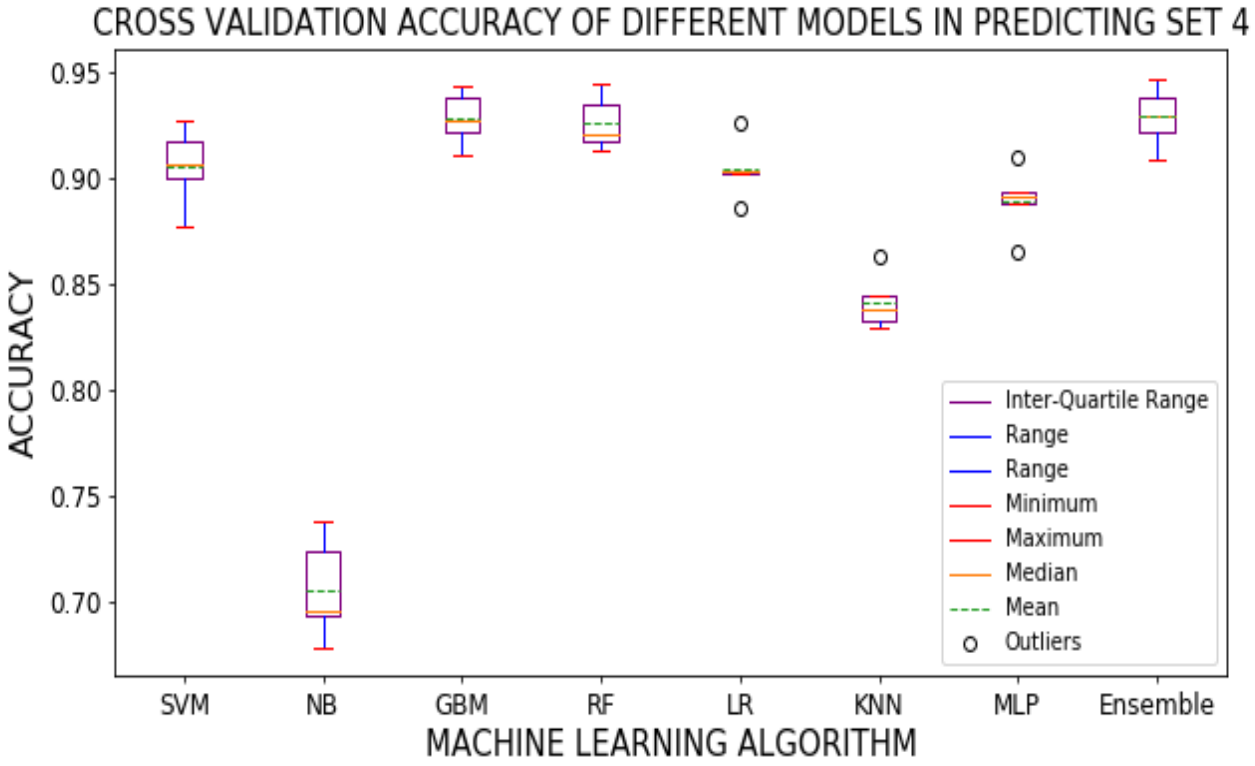

Fig. 8. Cross validation accuracy of the different machine learning algorithms on set 2 represented with a box plot. The mean cross validation scores of the ensemble and that of the GBM are equal. They both give the best performance while Naïve Bayes gives the worst performance.

second again and Naive Bayes still performs worse than the others. When the ensemble classifier is considered, it proves the best with the highest mean cross validation accuracy and a test accuracy equal to that of the best individual model

Overall, the ensemble classifier gives the best results in all the 4 problem sets but the difference is more significant for the 10 class (morphological pattern) classification problems.

VII. CONCLUSION

Of the machine learning algorithms used, ensembles such Gradient Boosting and Random Forests proved to be the best for all 4 classification sets. The results obtained proved better than those of other researchers who solved the classification problem using the same UCI data set. An ensemble of the best Gradient Boosting, Random Forest and the support vector machine models proved even better especially for the morphological pattern classification (10 class). This suggests that ensembles are better suited for CTG classification problems. Also, as expected, predicting the morphological pattern assuming the fetal state is known and predicting the fetal state when the morphological pattern is known gave better accuracies than the original predictions. This is attributed to the correlation between both labels.

REFERENCES

[1] D. Ayres de Campos et al. "SisPorto 2.0 A Program for Automated Analysis of Cardiotocograms," J Matern Fetal Med, 2000, vol. 5, pp. 311-318..

[2] P. A. Warrick, E. F. Hamilton, R. E. Kearney, and D. Precup, "A machine learning approach to the detection of fetal hypoxia during labor and delivery," Proceedings of the Twenty-Second Innovative Applications of Artificial Intelligence Conference, 2010, pp. 1865-1870.

[3] Z. Comert and A. F. Kocamaz, "Comparison of machine learning techniques for fetal heart rate classification," Special issue of the 3rd International Conference on Computational and Experimental Science and Engineering, 2017, vol. 132, pp. 451-454.

[4] C. Sundar, M. Chitradevi, and G. Geetharamani, "Classification of cardiotocogram data using neural network based machine learning technique," International Journal of Computer Applications, 2012, vol. 47, pp. 19-25.

[5] M. Arif, "Classification of cardiotocograms using Random Forest classifier and selection of important features from cardiotocogram signal," Biomaterials and Biomedical Engineering, 2015, vol. 2, pp. 173-183.

[6] T. Fushiki, "Estimation of prediction error by using K-fold cross-validation. Statistics And Computing," Springer US, 2009, vol. 21, pp. 137-146.

[7] J. Schneider, "Cross Validation," 1997, Available at: http://www.cs.cmu.edu/~schneide/tut5/node42.html

[8] S. A. Sontakke, J. Lohokare, R. Dani and P. Shivagaje, "Classification of Cardiotocography Signals Using Machine Learning," Intelligent Systems and Applications, 2018, pp. 439–450.

[9] D. Ayres-de-Campos, C.Y. Spong, E. Chandraharan, "FIGO consensus guidelines on intrapartum fetal monitoring: Cardiotocography," In J. Gynecol. Obstet. 2015, vol. 131, pp. 13–24.

[10] RS. Kamath , RK. Kamat, "Modelling fetal morphologic patterns through cardiotocography data: A random forest based approach," J Pharm Biol Chem Sci, 2016, vol. 7.

[11] P. Tomas , J. Krohova, P. Dohnalek, P. Gajdos. "Classification of cardiotocography records by random forest," 2016.

[12] R. A. R. S. Baluz, C. N. D. Santos, "Applying machine learning approaches to assess cardiotocography exams," Information Systems and Technologies (CISTI), 2011 6th Iberian Conference, 2011, pp. 1-6.

[13] Chamidah N, Wasito I., "Fetal state classification from cardiotocography based on feature extraction using hybrid K-Means and support vector machine," International Conference on Advanced Computer Science and Information Systems. IEEE, 2015, pp. 37-41.

[14] S. Das, K. Roy, C. K. Saha, "Fuzzy Membership Estimation Using ANN: A Case Study in CTG Analysis," 2014, pp. 221-228.

[15] L. Alkema, D. Chou, D. Hogan, S. Zhang, A. Moller and A. Gemmill, "Global, regional, and national levels and trends in maternal mortality between 1990 and 2015, with scenario-based projections to 2030: a systematic analysis by the UN Maternal Mortality Estimation Inter-Agency Group," 2015, vol. 387, pp. 462-474.

[16] United Nations, World Fertility Patterns, 2015. Available at https://www.un.org/en/development/desa/population/publications/pdf/fertility/world-fertility-patterns-2015.pdf

[17] J. H. Miao, K. H. Miao, "Cardiotocographic Diagnosis of Fetal Health based on Multiclass Morphologic Pattern Predictions using Deep Learning Classification," 2018.

[18] UNICEF Maternal Health. Available at https://data.unicef.org/topic/maternal-health/maternal-mortality

[19] C. J. Murray, "Global, regional, and national age-sex specific all-cause and cause-specific mortality for 240 causes of death, 1990-2013: a systematic analysis for the global burden of disease study 2013," Lancet, 2015, vol. 385, pp. 117-171.

[20] R. Neiger, "Long-Term Effects of Pregnancy Complications on Maternal Health: A Review," J Clin Med., 2017, vol. 6, no. 8, pp. 76.

[21] I. Ingemarsson, "Fetal monitoring during labor," Neonatology, 2009, vol. 95.

[22] FIGO News, "Report of the FIGO study group on the assessment of new technology: evaluation and standardization of fetal monitoring," Organized by G. Rooth, A. Huch, and R. Huch, International Journal of Gynecology & Obstetrics, 1987, vol. 25, pp. 159-167.

[23] I.C Stylios, V. Vlachos and I. Androulidakis, "Performance Comparison of Machine Learning Algorithms for Diagnosis of Cardiotocograms with Class Inequality", 2014.

[24] Z. Zhao, Y. Zhang, and Y. Deng, "A Comprehensive Feature Analysis of the Fetal Heart Rate Signal for the Intelligent Assessment of Fetal State," *Journal of clinical medicine*, 2018, vol *7*, no. 8, pp. 223.

[25] S. Jacob, G. Ramani, "Evolving Efficient Classification Rules from Cardiotocography Data through Data Mining Methods and Techniques," 2012, pp.78.

[26] M. Jeweski, R. Czabanski and J. Leski, "The influence of cardiotocogram Signal feature selection method on fetal state assessment efficacy," Journal of Medical Informatics and Technologies. 2014, vol. 23.

[27] RS. Kamath , RK. Kamat, "Modeling fetal morphologic patterns through cardiotocography data: Decision tree-based approach," Journal of Pharmacy Research. 2018, vol. 12.

[28] H. Wang et al. "CTG graph classification based on deep learning and ensemble classification" International Journal of Science, 2017, vol. 4.

[29] C. Cortes, V. Vapnik, "Support-Vector Networks," Kluwer Academic Publishers, 1995, pp. 273–297.

[30] N. Cristianini, J. Shawe-Taylor, "An introduction to support Vector Machines: and other kernel-based learning methods," Cambridge University Press, New York, NY, USA, 2000.

[31] V. N. Vapnik, "Statistical Learning Theory," 1998.

[32] S. Taheri, M. Mammadov, "Learning the naive bayes classifier with optimization models," Int. J. Appl. Math. Comput. Sci., 2013, vol. 23, pp. 787–795.

[33] L. Breiman. "Random Forests," Kluwer Academic Publishers, 2001, vol. 45, no. 1, pp. 5–32.

[34] A. Sarica, , A. Cerasa and A. Quattrone, "Random Forest Algorithm for the Classification of Neuroimaging Data in Alzheimer's Disease: A Systematic Review," Frontiers in aging neuroscience, 2017, vol. 9, pp. 329.

[35] L. Breiman. "Arcing The Edge," *Technical Report 486*. Statistics Department, University of California, Berkeley, 1997.

[36] J. H. Friedman. "Greedy Function Approximation: A Gradient Boosting Machine," 1999.

[37] T. Cover and P. Hart, "Nearest neighbor pattern classification," in IEEE Transactions on Information Theory, 1967, vol. 13, 21-27.

[38] J. Starkweather and A. Kay Moske, "Multinomial Logistic Regression," 2011.

[39] S. K. Pal and S. Mitra, "Multilayer perceptron, fuzzy sets, and classification," in IEEE Transactions on Neural Networks, 1992, vol. 3, pp. 683-697.

APPENDIX I

This appendix includes some samples of the source code for this project and the output generated.

```
from sklearn.ensemble import GradientBoostingClassifier

def GradBoost(Xtrain, ytrain, Xtest, ytest):
    clf = GradientBoostingClassifier()

    #doing grid search for best parameters with a 5-fold cross validation
    parameters = {'max_depth':[2,3],
                  'n_estimators':[81,82,85],
                  'min_samples_split':[3,4],
                  'subsample':[0.9,1],
                  'learning_rate':[0.2],
                  'min_samples_leaf':[1,2,3,5]}
    clf = GridSearchCV(clf, parameters, cv=5)

    #fitting the data
    clf.fit(Xtrain,ytrain)

    #get an array of the cross validation scores for the best parameters
    cv_acc = cross_val_score(clf, Xtrain, ytrain, cv=5 )

    # Predicting the test labels
    y_pred = clf.predict(Xtest)


    #creating confusion matrix and plotting it
    confusion_mtx = confusion_matrix(ytest, y_pred)
    plot_confusion_matrix(confusion_mtx, [str(i) for i in range(1,len(confusion_mtx)+1)])

    accuracy = (confusion_mtx.diagonal().sum()/confusion_mtx.sum())*100

    return print(classification_report(y_pred = y_pred, y_true = ytest),
                 '\n', 'cross validation accuracy =',
                 cv_acc, '\n', 'test accuracy =',
                 accuracy)
```

Fig. A1. Gradient Boosting grid search and cross validation code

```
from sklearn.model_selection import cross_val_score
from sklearn.svm import SVC
from sklearn.ensemble import RandomForestClassifier
from sklearn.ensemble import GradientBoostingClassifier
from sklearn.ensemble import VotingClassifier

#creating the gradient boosting classifier with the best parameters
clf1 = GradientBoostingClassifier(n_estimators=82,
                                  subsample=1,
                                  max_depth=3,
                                  min_samples_split=3,
                                  min_samples_leaf=3)

#creating the random forest classifier with the best parameters
clf2 = RandomForestClassifier(n_estimators=800, max_features='sqrt', max_depth=90, min_samples_split=5 )

#creating support vector classifier with the best parameters
clf3 = SVC(C=200, gamma=0.01, probability=True)

#creating the voting classifier with soft voting
eclf = VotingClassifier(estimators=[('lr', clf1), ('rf', clf2), ('gnb', clf3)], voting='soft', weights=[4,2,1

#Computing cross validation scores for all the models with their mean and standard devaition
for clf, label in zip([clf1, clf2, clf3, eclf], ['Gradient Boosting', 'Random Forest', 'SVM', 'Ensemble']):
    scores = cross_val_score(clf, X_train1, y_class_train, cv=5, scoring='accuracy')
    print("Accuracy: %0.5f (+/- %0.2f) [%s]" % (scores.mean(), scores.std(), label), scores)
```

```
Accuracy: 0.90168 (+/- 0.02) [Gradient Boosting] [0.91860465 0.9122807  0.86176471 0.92920354 0.88656716]
Accuracy: 0.89459 (+/- 0.02) [Random Forest] [0.91569767 0.89473684 0.87941176 0.91445428 0.86865672]
Accuracy: 0.85989 (+/- 0.01) [SVM] [0.88081395 0.85964912 0.85588235 0.86430678 0.83880597]
Accuracy: 0.90169 (+/- 0.03) [Ensemble] [0.92732558 0.90350877 0.85882353 0.92625369 0.89253731]
```

Fig. A2. Ensemble voting classifier sample code

```
plt.figure(figsize=(10,5))

#create a box plot of the cross validation results of every model
plt.boxplot([cv1_svm, cv1_NB, cv1_GB, cv1_RF, cv1_LR, cv1_KNN, cv1_MLP, cv1_Ensemble],
            boxprops={'color':'purple'},
            whiskerprops={'color':'blue'},
            capprops={'color':'red'},
            showmeans=True,
            widths=0.25,
            meanline=True)

plt.xlabel("MACHINE LEARNING ALGORITHM", fontsize=15)
plt.ylabel('ACCURACY', fontsize=15)
plt.title('CROSS VALIDATION ACCURACY OF DIFFERENT MODELS IN PREDICTING SET 1', fontsize=15)

#get current axis
ax = plt.gca()

#rename the xtick labels and change the font size
c=['SVM','NB','GBM','RF','LR','KNN','MLP', 'Ensemble']
ax.set_xticklabels(c)
ax.tick_params(axis=u'both', which=u'both',length=3, labelsize=12)

#create a legend
ax.legend(['Inter-Quartile Range', 'Range', 'Range','Minimum','Maximum','Median','Mean','Outliers'])
plt.savefig('set 1 cross validation.png')
```

Fig. A3. Sample box plot code