



**TRACKING AND OBSERVING THE FETAL MOVEMENTS BY USING
DEEP LEARNING METHODS**

Haad AKMAL

**IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF PhD THESIS
IN ELECTRICAL ELECTRONICS ENGINEERING DEPARTMENT**

**GAZI UNIVERSITY
GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES**

FEBRUARY 2024

ETHICAL STATEMENT

I hereby declare that in this thesis study I prepared in accordance with thesis writing rules of Gazi University Graduate School of Natural and Applied Sciences;

- All data, information and documents presented in this thesis have been obtained within the scope of academic rules and ethical conduct,
- All information, documents, assessments and results have been presented in accordance with scientific ethical conduct and moral rules,
- All material used in this thesis that are not original to this work have been fully cited and referenced,
- No change has been made in the data used,
- The work presented in this thesis is original

or else, I admit all loss of rights to be incurred against me.

Haad AKMAL

02/02/2024

TRACKING AND OBSERVING THE FETAL MOVEMENTS BY USING DEEP
LEARNING METHODS

(Ph. D. Thesis)

Haad AKMAL

THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES OF
GAZİ UNIVERSITY

February 2024

ABSTRACT

The main objective of this thesis is to provide solutions to observe and track the fetal movement (hence fetal condition) with the help of machine learning. This thesis uses Cardiotocography (CTG) and Fetal Electrocardiogram (FECG) datasets. The thesis has three phases such that Phase A and Phase B utilize CTG whereas Phase C utilizes FECG. The CTG dataset contains data related to Fetal heart rate (FHR) and uterine contractions (UC), which are used to evaluate the condition of the fetus during labor or the third trimester. In order to categorize fetal conditions and CTG morphological patterns, Phase A proposes a deep learning model that corrects the CTG dataset imbalance and incorporates an autoencoder for feature extraction, recursive feature elimination for feature selection, and Bayesian optimization to further fine tune the model. The proposed model classified fetal condition with 96.62% accuracy and CTG morphological pattern with 94.96% accuracy. In Phase B a survey of all the relevant classifiers for CTG dataset resulted in Random Forest being selected for the proposed model, that utilizes feature elimination and hyperparameter optimization for classifying the fetal condition. The proposed model displayed an accuracy of 97.20%. In Phase C Long Short-Term Memory (LSTM) neural network was used to predict FECG signals and then detect the anomalies in it as well. The study shows that LSTM-based model is highly accurate in predicting FECG signals and spotting abnormalities. The proposed model could improve the accuracy of FECG analysis, reducing the incidence of fetal discomfort and prenatal death rates.

Science Code	: 90542
Key Words	: Diagnostics, Cardiotocography, Fetal heart rate, Fetal wellbeing, Machine learning, Classification, Feature extraction, Feature selection, Bayesian optimization, Fetal electrocardiogram, Anomaly detection, Artificial neural networks
Page Number	: 116
Supervisor	: Prof. Dr. Fırat HARDALAÇ

DERİN ÖĞRENME YÖNTEMLERİNİ KULLANARAK FETAL HAREKETLERİ İZLEME VE GÖZLEMLEME

(Doktora Tezi)

Haad AKMAL

GAZİ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

Şubat 2024

ÖZET

Bu tezin temel amacı, makine öğrenimi yardımıyla fetal hareketin (dolayısıyla fetal durumun) gözlemlenmesi ve izlenmesi için çözümler sunmaktır. Bu tez Kardiyotokografi (CTG) ve Fetal Elektrokardiyogram (FECG) veri setlerini kullanır. Tez, Bölüm A ve Bölüm B'nin CTG kullandığı, Bölüm C'nin ise FECG kullandığı üç duruma sahiptir. CTG veri seti, doğum eylemi veya üçüncü trimester sırasında fetüsün durumunu değerlendirmek için kullanılan Fetal kalp hızı (FHR) ve uterus kasılmaları (UC) ile ilgili verileri içerir. Fetal durumları ve CTG morfolojik modellerini kategorize etmek için, Bölüm A, CTG veri seti dengesizliğini düzelter ve özellik çıkarımı için bir otomatik kodlayıcı, özellik seçimi için özyinelemeli özellik eleme ve modelde daha fazla ince ayar yapmak için Bayes optimizasyonu içeren bir derin öğrenme modeli önerir. Önerilen model, fetal durumu %96,62 doğrulukla ve CTG morfolojik modelini %94,96 doğrulukla sınıflandırmıştır. Bölüm B'de, CTG veri kümesi için ilgili tüm sınıflandırıcıların bir araştırması, fetal durumu sınıflandırmak için özellik eleme ve hiperparametre optimizasyonunu kullanan önerilen model için Random Forest'ın seçilmesiyle sonuçlandı. Önerilen model, %97,20'lik bir doğruluk göstermiştir. Bölüm C'de, FECG sinyallerini tahmin etmek ve ardından ondaki anormallikleri de tespit etmek için Uzun Kısa Süreli Bellek (LSTM) sinir ağı kullanıldı. Çalışma, LSTM tabanlı modelin FECG sinyallerini tahmin etmede ve anormallikleri tespit etmede oldukça doğru olduğunu gösteriyor. Önerilen model, FECG analizinin doğruluğunu artırabilir, fetal rahatsızlık insidansını ve doğum öncesi ölüm oranlarını azaltabilir.

Bilim Kodu : 90542

Anahtar Kelimeler : Teşhis, Kardiyotokografi, Fetal kalp atış hızı, Fetal sağlık, Makine öğrenimi, Sınıflandırma, Özellik çıkarımı, Özellik seçimi, Bayes optimizasyonu, Fetal elektrokardiyogram, Anomali tespiti, Yapay sinir ağları

Sayfa Adedi : 116

Danışman : Prof. Dr. Fırat HARDALAC

ACKNOWLEDGEMENT

To everyone who has helped to complete my thesis, I would like to offer my sincere gratitude and admiration. This path has been one of intellectual and personal improvement, and I am appreciative of the encouragement and support that have seen me through. I praise ALLAH Almighty for blessing me with regards to finishing this thesis. I want to start off by sincerely thanking my supervisor, Prof. Dr. Fırat HARDALAÇ, for all of his help, patience, and unshakable faith in the potential of this effort. His efforts, helpful criticism, and insights have been crucial in determining the course of this thesis. I would like to express my gratitude to my parents, family and friends who have served as a source of motivation, steadfast support and inspiration for me along this journey of the thesis. Their companionship and readiness to participate in stimulating talks have really aided in the development of my concepts. I also owe a debt of gratitude to my department professors for their vast knowledge and dedication to providing a dynamic academic atmosphere. My comprehension has been strengthened by their lectures, conversations, and interactions, giving me a strong basis for my project. I especially want to thank Mr. Kubilay AYTURAN for providing support throughout the duration of the thesis. Last but not least, I would like to express my sincere gratitude to all the researchers and academics whose works served as the cornerstone for my thesis.

LIST OF CONTENTS

	Page
ABSTRACT.....	iv
ÖZET	v
ACKNOWLEDGEMENT	vi
LIST OF CONTENTS	vii
LIST OF TABLES	x
LIST OF FIGURES	xi
SYMBOLS AND ABBREVIATIONS.....	xiii
1. INTRODUCTION.....	1
2. RELATED WORKS.....	9
3. RESEARCH DESIGN AND METHODOLOGICAL FRAMEWORK .	17
3.1. Overview of Classification.....	17
3.1.1. K-nearest neighbors (KNN)	18
3.1.2. Multi-layer perceptron (MLP).....	19
3.1.3. Support vector machine (SVM)	19
3.1.4. Decision tree (DT).....	20
3.1.5. Random forest (RF).....	21
3.1.6. Feature importance.....	22
3.1.7. Long short-term memory (LSTM)	24
3.2. Feature Extraction	26
3.2.1. Autoencoder	27
3.3. Feature Selection	29
3.3.1. Recursive feature elimination	31
3.4. Overview of Hyperparameter Optimization.....	33
3.5. Synthetic Minority Oversampling Technique	36

	Page
3.6. Overview of Performance Analysis	38
3.6.1. Confusion matrix.....	39
3.6.2. Receiver operating characteristic curve	40
3.6.3. Precision recall curve	41
3.6.4. SHapley additive exPlanations (SHAP) analysis	42
3.6.5. Matthew's correlation coefficient.....	44
3.6.6. Mean squared error (MSE)	45
3.7. Dataset Overview	45
3.7.1. Correlation analysis.....	49
3.7.2. Data preprocessing	52
3.7.3. Dataset for Phase C	54
3.8. Methodology	56
3.8.1. Methodology for Phase A	56
3.8.2. Methodology for Phase B.....	59
3.8.3. Methodology for Phase C.....	60
4. RESULTS.....	63
4.1. Results of Phase A	63
4.1.1. Fetal status classification in Phase A	63
4.1.2. CTG Morphological Pattern Classification in Phase A	66
4.1.3. Bayesian optimization in Phase A.....	70
4.1.4. Overview of SHAP analysis in Phase A	72
4.2. Results of Phase B.....	75
4.2.1. Control Setup in Phase B	75
4.2.2. Feature Elimination in Phase B.....	76
4.2.3. Hyperparameter optimization in Phase B	81
4.3. Results of Phase C.....	82

	Page
4.3.1 FECGSYN Case 1 of Phase C	82
4.3.2. FECGSYN Case 2 of Phase C	85
4.3.3. FECGSYN case 4 of Phase C	87
4.3.4. Performance Analysis of Phase C	89
5. DISCUSSION	91
6. CONCLUSION.....	105
REFERENCES	107
CURRICULUM VITAE.....	115

LIST OF TABLES

Table	Page
Table 3.1. Metrics Used for Measuring the Performance	39
Table 3.2. The Features or Predictors in the CTG Dataset	46
Table 3.3. The CTG Related Classes of Morphological Patterns in the CTG Dataset ...	47
Table 3.4. FECGSYN Dataset Overview	55
Table 4.1. Performance Evaluation of the Proposed Fetal Status Model	64
Table 4.2. Performance Evaluation of RF (without PM) for Fetal Status	64
Table 4.3. Confusion Matrix of The Proposed Fetal Status Model	64
Table 4.4. Confusion Matrix of Basic RF for Fetal Status	64
Table 4.5. Performance Evaluation of the Proposed CTG Morphological Pattern	67
Table 4.6. Performance Evaluation of RF (without PM) for CTG Morphology	67
Table 4.7. Confusion Matrix of The Proposed CTG Morphological Pattern Model	68
Table 4.8. Confusion Matrix of basic RF for CTG Morphological Pattern	68
Table 4.9. Performance Evaluation of the Proposed Fetal Status Model, without BO...	71
Table 4.10. Confusion Matrix of the Proposed Fetal Status Model, without BO	71
Table 4.11. Performance Comparison of Classifiers in Part 1 (all 21 features)	76
Table 4.12. Confusion Matrix of The Best Classifier (RF) in Part 1 (all 21 features) ...	76
Table 4.13. Performance Comparison of Classifiers in Part 2B (17 features)	78
Table 4.14. Confusion Matrix of The Best Classifier (RF) in Part 2B (17 features)	79
Table 4.15. Performance Analysis of The Proposed Model in Phase B (11 features)	82
Table 4.16. Confusion Matrix of The Proposed Model in Phase B (11 features)	82
Table 4.17. Performance Indicators and Threshold Relation with Anomalies	89
Table 5.1. Comparing this study (Phase A and Phase B) with relevant literature	95

LIST OF FIGURES

Figure	Page
Figure 3.1. Imbalance Characteristics of the CTG Dataset in the Fetal Status Case	48
Figure 3.2. Imbalance Characteristics of the CTG Dataset in the Morphological Pattern Classes	48
Figure 3.3. Correlation Heatmap of the CTG Dataset using Spearman Correlation	50
Figure 3.4. Correlation Heatmap of the CTG Dataset using Kendall Correlation.....	51
Figure 3.5. FECGSYN Generation [20]	55
Figure 3.6. Proposed Algorithm for Phase A.....	57
Figure 3.7. Flowchart of the Proposed Algorithm for Phase C	61
Figure 4.1. ROC Curve (AUC) of the Proposed Fetal Status Model.....	65
Figure 4.2. PR Curve of the Proposed Fetal Status Model	65
Figure 4.3. Relationship between Model Accuracy and the Total Number of Features Selected for Fetal Status in Phase A	66
Figure 4.4. ROC Curve (AUC) of the Proposed CTG Morphological Pattern Model ...	69
Figure 4.5. PR Curve of the Proposed CTG Morphological Pattern Model.....	69
Figure 4.6. Relationship between Model Accuracy and the Total Number of Features Selected for CTG Morphological Pattern in Phase A.....	70
Figure 4.7. The Fetal Status SHAP Analysis Summary Plot.....	73
Figure 4.8. The CTG Morphological Pattern SHAP Analysis Summary Plot	74
Figure 4.9. Importance of Features in CTG Dataset via XGBoost.....	77
Figure 4.10. Importance of Features in CTG Dataset via RF	77
Figure 4.11. ROC Curve (AUC) of RF in Part 1 (with 21 features).....	79
Figure 4.12. ROC Curve (AUC) of RF in Part 2 (with 17 features).....	80
Figure 4.13. Effect of Training Sample Size on Model Accuracy.....	81
Figure 4.14. Case 1 FECGSYN Signal.....	83
Figure 4.15. Case 1 Model Loss	83
Figure 4.16. Case 1 Predicted ECG Signal.....	84

Figure	Page
Figure 4.17. Case 1 Anomaly Detection via Proposed Model. Top Left to Right: 03 dB SNR, 06 dB SNR. Bottom Left to Right: 09 dB SNR, 12 dB SNR	84
Figure 4.18. Case 2 FECGSYN Signal.....	85
Figure 4.19. Case 2 Model Loss	85
Figure 4.20. Case 2 Predicted ECG Signal.....	86
Figure 4.21. Case 2 Anomaly Detection via Proposed Model. Top Left to Right: 03 dB SNR, 06 dB SNR. Bottom Left to Right: 09 dB SNR, 12 dB SNR	86
Figure 4.22. Case 4 FECGSYN Signal.....	87
Figure 4.23. Case 4 Model Loss	87
Figure 4.24. Case 4 Predicted ECG Signal.....	88
Figure 4.25. Case 4 Anomaly Detection via Proposed Model. Top Left to Right: 03 dB SNR, 06 dB SNR. Bottom Left to Right: 09 dB SNR, 12 dB SNR	89
Figure 5.1. NSP Disposition in the CLASS of Morphological Pattern.	93
Figure 5.2. Performance Comparison of Classifiers in Phase Part 1 (21 features) in Red and Part 2 (17 features) in Green.	97
Figure 5.3. Relation of Number of Features with Accuracy.....	99
Figure 5.4. Relation of Number of Features with MCC.	99
Figure 5.5. SHAP Analysis of CTU-UHB CTG Dataset.....	102

SYMBOLS AND ABBREVIATIONS

The symbols and abbreviations used in this study are presented below with their explanations.

Symbols	Explanations
---------	--------------

dB	Decibel
----	---------

Abbreviations	Explanations
---------------	--------------

SMOTE	Synthetic Minority Oversampling Technique
-------	---

SHAP	SHapley Additive exPlanations
------	-------------------------------

FIGO	International Federation of Gynecology and Obstetrics
------	---

FECG	Fetal Electrocardiogram
------	-------------------------

CTG	Cardiotocography
-----	------------------

APGAR	Appearance, Pulse, Grimace, Activity, and Respiration
-------	---

1. INTRODUCTION

The field of tracking and observing fetal movement has seen a remarkable transformation in the quest to ensure the safety of both mothers and their unborn offspring. In this quest, the fusion of medical knowledge with the strength of deep learning techniques has opened the door for ground-breaking developments. This investigation digs into how contemporary computational methods and obstetric care might work together, illuminating the transformational process of tracking fetal movements via the prism of advanced deep learning approaches. Join us as we set out on a journey that aims to improve prenatal care while also expanding our knowledge of the complex dance of life before delivery. In this study two methods are used. These methods are used to observe the fetal wellbeing. One method is cardiotocography (CTG) and the other method is fetal electrocardiogram (FECG).

CTG is a method that is used to check on the health of the fetus throughout the third trimester or during childbirth [1, 2]. A pressure transducer attached to the abdominal wall and an external ultrasound probe on the maternal abdominal wall are used by CTG to constantly monitor maternal uterine contractions (UC) and fetal heartbeats (also called fetal heartrate - FHR). CTG is a type of non-stress technique (NST). Other similar techniques that are not NST based are called invasive techniques. In such techniques a fetal scalp electrocardiographic electrode is implanted during delivery that can also be used for monitoring the fetal condition. In some CTG devices, real-time displays of the simultaneous readouts are also possible. That real-time display can be in the form of analog (electrocardiogram paper type print out) or digital (display in adjacent monitor). In assessing the fetal wellbeing, FHR plays a very important role. The balance of the fetal sympathetic and parasympathetic nervous systems is reflected by the fetal heart rate (FHR) [3], and variations from baseline rates in response to uterine contractions can indicate fetal distress from hypoxia and acidemia, which may call for therapeutic interventions, such as a hurried delivery. In healthy fetuses, the FHR response exhibits significant intra- and inter-observer variability as well as physiological heterogeneity. Although the specificity and positive predictive value of manual (without the use of a support system based on machine learning) CTG interpretation are poor [1, 2], the negative predictive value is good. During delivery, invasive fetal blood collection to confirm metabolic disturbances may help reduce the intervention rate for false positive CTG [1]. Clinicians commonly interpret CTG as Normal, Suspect, or Pathologic based on expert criteria [1]. CTG is one of the most widely used

methods for gauging and evaluating the fetal health (also called fetal wellbeing) in affluent nations [4]. Even some researchers claim that CTG is overutilized in low-risk circumstances [4]. Given that an abnormal (also called pathological) CTG result is associated with a poor APGAR (Appearance, Pulse, Grimace, Activity and Respiration) score and neonatal intensive care units (NICU), there is a correlation between CTG and perinatal mortality and morbidity [5]. Fetal distress can also be assessed by noting the status or condition of fetus. Fetal discomfort can lead to a range of outcomes, depending on the underlying reasons, the severity of the distress, and how quickly medical treatments are made. A fetus exhibiting symptoms of compromised wellness or distress during pregnancy or labor is referred to as being in fetal distress. It is a sign that the fetus is not getting enough oxygen or nutrients, which could hurt it or have negative effects if it is not treated right away. If the fetal distress is momentary, the posture of mother can be changed, oxygen can be given to the mother, intravenous fluids can be adjusted, or if necessary, near the end of the third trimester an emergency cesarean section can be performed. All of these actions have the potential to benefit the fetal status and have favorable results after labor. Fetal distress (if it continues for a longer period of time), however, can result in long-term detrimental effects including cognitive impairments, learning difficulties, motor impairments, diseases like cerebral palsy, or even fetal mortality (in rare circumstances). Long-lasting fetal discomfort is typically caused by an oxygen shortage [6]. Birth asphyxia, which is the primary cause for almost 900,000 newborn fatalities yearly, can also occur due to long term fetal distress [7]. Fetal mortality is more prevalent generally in low-income countries than in high-income countries, this fact highlights the disparities in healthcare resources and access across these regions. Although the worldwide newborn death rate (per 1000 live births) has fallen over the past three decades, from 36.7 in 1990 to 17 in 2020, it is still significantly higher in low-income areas [8]. Placental problems, which are often connected to fetal distress, were one of the most frequent causes of fetal mortality even in high-income areas. In 2020, placental problems and issues caused 24.4% of fetal fatalities in the District of Columbia, USA [9]. In order to evaluate the fetal wellbeing, it is crucial to understand the condition of the fetus. The CTG can detect fetal discomfort early on during pregnancy. CTG tests minimize patient pain since they are time and resource efficient, especially when the number of patients is large. A subject usually goes to a clinic or hospital to get a CTG test. However, considering the risk of recent COVID pandemic (which is still present on a smaller scale) or a future pandemic, the stay-at-home type methods to implement CTG with machine learning are quite popular. Even government health officials [10] are interested in such a system to non-

invasively implement CTG using a machine learning system. In such a system CTG data can be transmitted to cloud services that would be linked to machine learning based systems that could diagnose and provide decision support system to the medical staff in a remote location. This system can reduce resources like time and human error. Moreover, by decreasing the time costs of manual interpretation and limiting human error, intra- and inter-observer variability, machine learning-enabled automated computer-aided diagnosis of CTG can serve as a valuable objective approach for high-throughput real-time surveillance of fetal wellbeing [11]. According to research in [12,13] fixed FHR baselines, a loss of FHR variability, and the absence of accelerations in CTG tracing patterns are signs of a troubling situation for a fetus. For CTG phase, an expert interprets CTG visually, and automated procedures are suggested to support this process. While considerable research [14-17] has also been done in utilizing machine learning in assessing the fetal condition and fetal hypoxia. Although it has not yet been demonstrated that machine learning-based CTG interpretation significantly improves pregnancy outcomes when compared to expert interpretation [17], their use cases as screening and decision-support tools are still valid given the substantial volume of CTG data that is anticipated to be produced by intermittent or continuous remote monitoring and that would otherwise all need to be manually interpreted by specialists. The clinical necessity for precise computer-aided diagnosis of CTG signals in real time is pressing. One of the most important issues related to computer aided diagnosis of CTG is the missing data issue. Research in [18] describes a technique for filling in missing data that uses two iterative phases, the issue is discussed in detail later data preprocessing section.

Another method used in this study is fetal electrocardiogram (FECG). FECG is used to track the electrical activity of fetal heart throughout pregnancy. Similar to the adult heart, the fetal heart generates electrical impulses when it contracts and relaxes. Monitoring these signals can yield important knowledge about the fetal health and wellbeing. FECG can be observed using one of following two techniques:

In direct FECG, the electrodes are directly applied to the fetal scalp. It is frequently used to monitor the FHR more precisely and constantly throughout labor and delivery. Compared to other techniques, direct FECG can offer a more thorough and precise picture of the electrical activity of the fetal heart.

In indirect FECG (also called abdominal FECG), the electrodes are applied to the belly of mother. These electrodes (or sensors) detect the electrical activity of the fetal heart as they pass through the amniotic fluid and the maternal tissues. The FHR and rhythm are then determined by processing the collected information. Indirect FECG is also a non-invasive technique.

The usage of FECG during pregnancy consists of the following aspects:

To monitor the fetus. During labor and delivery, FECG is frequently used to keep an eye on the fetal heart rate and rhythm. It aids medical professionals in identifying any variations in the fetal heartbeat that could be signs of distress or other possible problems.

To assess the fetal health. The heart rate and rhythm of the fetus can be continuously monitored to get information about fetal health. Changes in the heart rate pattern may be a sign of issues including oxygen shortage or fetal discomfort.

To assess the maternal health. Monitoring the fetal heart rate might also, inadvertently, provide information about the health of the mother. Certain maternal illnesses, such diabetes or hypertension, might have an impact on the health of the fetus and may show up in the fetal heart rate patterns.

To provide opportunities to do research and analysis in this field. FECG data can be useful for research, enabling specialists in medicine and science to examine the growth, behavior, and reactions of the fetal heart.

To provide preventive care. In some situations, FECG can assist medical professionals in identifying potential issues early in pregnancy and taking the necessary steps to control them, assuring the best results for both the mother and the fetus.

The importance of FECG analysis in prenatal care is summarized as follows:

Anomalies in FECG signals can indicate potential fetal distress.

Lowering antenatal mortality rates through proactive monitoring and intervention.

FECG analysis provides valuable insights into fetal wellbeing and helps guide medical decisions.

Improving the quality of FCG analysis can lead to better management of pregnancy complications.

Timely detection of anomalies allows for appropriate interventions to prevent adverse outcomes.

While FCG can offer useful information, it is usually evaluated in conjunction with other clinical data and evaluations, it is crucial to keep in mind. In order to make well-informed decisions about patient treatment, healthcare professionals are educated to evaluate the FCG data and take into account the larger context of the pregnancy. Fetal electrocardiography, in general, plays a significant role in maternal-fetal medicine by providing a window into the electrical activity of the fetal heart, assisting in the evaluation of fetal wellbeing, and promoting safer pregnancies and births.

This thesis study has been divided into three phases, namely Phase A, Phase B and Phase C. Here Phase A and Phase B focus on CTG whereas Phase C focuses on FCG. Two different datasets are used in this thesis. This was done to make a better system utilizing both datasets to verify the proposed models. The net objective is to have a system of tools that observe fetal wellbeing and provide a knowledge base that uses two different datasets for tracking and observing the fetal movements by using deep learning methods.

The hypothesis for Phase A in this study is to determine whether it is possible to accurately diagnose and categorize the various fetal status (called normal, suspect, pathologic in this study), as well as the CTG morphological patterns (ten different pattern in this study), using a machine learning (deep learning) model based on feature extraction, feature selection, and Bayesian optimization. This research could provide a decision support tool for managing pregnancies. The following is the explanation of the objectives proposed for Phase A in this study:

- The most important object is to diagnose fetal wellbeing. This objective is achieved first by addressing the imbalance CTG dataset [19].

- Second objective is to make an encoder-bottleneck information coefficient is proposed that is then used to implement a feature extraction (FE) module. The primary aim of the FE module is to address the relatively larger size of the CTG dataset achieved after addressing the imbalanced nature in the first objective.
- Third objective is to integrate the proposed model with a feature selection (FS) module. The primary aim of the FS module is to further remove the redundant information in the CTG dataset.
- Fourth objective is to further fine tune the performance of the proposed model by implementing Bayesian optimization.
- The underlying objective of Phase A is to combine all the above-mentioned modules and finally to integrate classification ability in the proposed model as well.

The hypothesis for Phase B in this study is to find a suitable decision support system that classifies the fetal conditions using machine learning based model utilizing feature elimination and hyperparameter optimization. The following is the explanation of the objectives proposed for Phase B in this study:

- One objective is to perform a survey study of all the relevant and popular classifiers for the CTG dataset [19] and find out the best classifier.
- Second objective is to find feature importance of the selected CTG dataset that can then be used for feature elimination.
- The features are removed based on a proposed model that utilizes the values of Matthew's correlation coefficient (MCC) and recursive feature elimination (RFE).
- After redundant features are removed then the fourth objective is to further improve the performance of the best classifier by feeding the model into a hyperparameter optimization module.

The hypothesis for Phase C in this study is related to this research statement: Can Long Short-Term Memory (LSTM) be used to predict synthetic FECG time series signals and detect anomalies in them? The following is the explanation of the objectives proposed for Phase C in this study:

- First objective is to preprocess the synthetic FECG dataset [20] into a "neural network suitable" form.
- Second objective is to construct an LSTM model that is suitable for the given FECGSYN dataset [20].

- Third objective is to apply an anomaly detection condition to the trained LSTM model.
- Finally, analysis of the results based on the selected performance metrics is performed to judge the quality of the proposed model for this Phase C.

The outline of the thesis is as follows. In section 2 (Related works) related and relevant literature review is given. In section 3 (Research Design and Methodological Framework) overview regarding the modules utilized in the research study is given. Moreover, relevant mathematical formulae along with the overview of two datasets and overview of performance metrics used in this study are given in this section. Additionally, all three proposed algorithms are given in sub section 3.8 (Methodology). All the results, graphs and tables are provided in section 4 (Results). Detailed discussion about the results is given in section 5 (Discussion). Moreover, how the proposed algorithms performed with respect to the objectives proposed in section 1 (Introduction) are also given there. In section 6 (Conclusion), the whole research is concluded. Moreover, recommendations for future work are also given in that section. Lastly, all the references and bibliography are given in the References section.



2. RELATED WORKS

Gazi Fetal movement analysis (tracking and observing) is essential for determining the health and development of a developing fetus during pregnancy. As medical technology has developed over time, it has become easier to track and analyze these movements, which has enhanced prenatal care and allowed for the early identification of possible issues. Deep learning techniques have recently shown considerable promise in improving the precision and effectiveness of tracking and observing fetal movements. In this part, we provide a summary of the literature that is currently available on the monitoring and observation of fetal movements, highlighting both established methods and recent developments in the use of deep learning related machine learning techniques.

Machine learning can predict the target clinical conditions with high confidence [21].

A hybrid method was proposed in [22] that utilized the K-Means algorithm to perform feature extraction. The resulting dataset was then fed to a Support Vector Machine (SVM) to perform classification, which achieved 90.64% accuracy.

Authors in [23] outlined a mechanism to observe and read CTG via the Omniview-SisPorto 2.0 software. The resulting dataset would be discussed in detail in dataset overview section.

In [24] multiple algorithms were used on the CTG dataset [19], these algorithms were ANN (Artificial Neural Network), LSTM (Long Short-Term Memory), XGB (eXtreme Gradient Boosting), KNN (K-Nearest Neighbor), SVM (Support Vector Machine), RF (Random Forest) and LGBM (Light Gradient Boosting Machine). For all these algorithms, parameter optimization was performed using grid search. The following parameters were used to tweak in these algorithms. For ANN, the selected parameters for optimization were node in hidden layer, optimizer, activation function, and batch; For LSTM, the selected parameters for optimization were node in hidden layer, optimizer, activation function, and batch; For XGB, the selected parameters for optimization were minimum child weight, and subsample; For SVM, the selected parameters for optimization were C, and kernel function; For KNN, the selected parameters for optimization were number of neighbors; For LGBM, the selected parameters for optimization were number of leaves, and type of boosting; For RF, the selected parameters for optimization were minimum sample split, maximum features, and

number of estimators. Additionally, the authors used balancing techniques to see the results of the models. LGBM, KNN and RF performed better as compared to the other algorithms. In [25] the authors used multiple algorithms like KNN, Linear SVM, Naïve Bayes (NB), Decision Tree (DT), Ada Boost, Bagging and Stacking. Ada Boost with RF performed the best on CTG dataset [19] with an accuracy 94.7%.

In [26] multiple algorithms like DT, SVM, RF ANN and Gradient Boosting were used on the CTG dataset [19]. The authors manually selected top 16 features for classification by using Gradient Boosting. ANN with feature selection displayed the best results among selected classifiers. Results of the best classifiers namely DT, SVM and RF classifiers achieved accuracy rates of 90%, 93.41%, and 93.41%, respectively.

The authors in [27] also used the CTG dataset [19]. Using that CTG data, ten different machine learning classification models were developed. These algorithms were Multilayer Perceptron (MLP), XGBoost, DT, RF, Logistic Regression (LR), SVM Linear Kernel, SVM RBF Kernel, KNN, Naïve Bayes, AdaBoost. The performance metrics used to predict the normal, suspicious, and pathological fetal states were sensitivity, precision, F1 score, and total accuracy. Then, the model with the greatest performance on the given metrics was found. XGBoost classifier performed the best amongst all the other selected classifiers. XGBoost achieved an accuracy of 93% and that outperformed all other classifiers.

The research in [28] used relevant algorithms for CTG dataset [19], such as DT, SVM and NB. The authors found out that DT performed the best with an accuracy of 93.17%. The results of the research showed that DT, SVM and Naïve Bayes algorithms achieved accuracy rates of 93.17%, 92.84% and 83.65%, respectively, for the CTG dataset [19].

In [29] a model was proposed that was able to predict the fetal health condition. The authors also compared multiple classifiers Logistic Regression, KNN, RF and Gradient Boosting Machine (GBM) for the CTG dataset [19] and selected the best one in the proposed model. They evaluated the classifiers with respect to accuracy, precision, recall, and F1-score. The authors observed that RF performed the best. This study was conducted to determine the suitable model for the CTG case, in which Logistic Regression (LR), K-Nearest Neighbors (KNN, RF, and Gradient Boosting Machine (GBM) are used. The authors concluded that RF

is the quite suited for the CTG with an accuracy of 93%, followed by KNN (90%), GBM (90%), and LR (89%).

Authors in [30] proposed CART (Classification and Regression Tree) to be used with CTG dataset [19]. The goal of the research was to observe the fetal health status. In the research 5- fold cross validation was performed, and their model was analyzed using recall, F1-score and precision. Their model displayed 90.12% model accuracy. In the above-mentioned literature review articles, most of them had similar types of classifiers.

However, different types of algorithms were used in [31] such as Gradient Descent, Conjugate Gradient, Quasi-Newton, Resilient Backpropagation (RP), and Levenberg Marquardt (LM) algorithms. Here backpropagation (BP) is used to perform gradient in these algorithms. BP transfers the error between output and input backward towards the input, hence it is called backpropagation. Its aim is to optimize the weights so that varying inputs can be easily learnt by the neural network. Authors observed that RP algorithm performed the best as compared to others displaying an accuracy of 93.60%. When updating the weights, the heuristic learning method RP increased the speed of convergence by considering just the sign and not the size of the derivative of the error function.

Authors in [32] performed Dimensionality Reduction (DR) on CTG dataset [19]. Moreover, after DR, they performed classification using multiple algorithms like DT, NB, RF, and SVM. The final result of the proposed models was not outstanding in terms of accuracy; however, the idea to use dimensionality reduction techniques like Linear Discriminant Analysis (LDA), and Principal Component Analysis on the CTG dataset [19] was quite intriguing. Overall, a range of classifiers, including ANN, LSTM, XGB, SVM, KNN, LGBM, RF, Ada Boost, Bagging and Stacking, DT, NB, LR, CART, LM backpropagation, RP, and GBM, have been used in the aforementioned almost identical comparative studies [24–32] using the CTG dataset [19]. The studies have accuracy percentages between 83.65% and 96.61%. In general, general research regarding CTG dataset [19] has displayed that RF is the most effective classifier for this phase.

For the research in [33], the authors used the NB classifier as a base and coalesced with the Firefly algorithm. For feature selection RF was used. Their model achieved an accuracy of 86.54% (with 8 features selected). In this research a unique approach (that combined Firefly

algorithm with Naïve Bayes classifier) allowed certain CTG features to be randomly selected and the model executed in swarms of 10, 30 and 50. Model accuracy increased from 81.84% to 86.54% when all 21 features and only 8 features were selected, respectively. Swarm size of 50 gave the best results but had the disadvantage of being more time-consuming.

In [34], the authors utilized a stacked model method. As the CTG dataset [19] is imbalanced in nature, thus in this method a mixture of multiple models was used. This stacked model was preferred for its anti-interference traits. Their model displayed 96.08% model accuracy. That stacked model had a combination of multiple models, like LR, SVM, BP (Back Propagation), XGBoost, RF and, GBDT (Gradient Boosting Decision Tree).

For the CTG dataset [19], Synthetic Minority Oversampling Technique (SMOTE) was implemented along with an AutoML method in [35]. In this research SMOTE countered the imbalance dataset. PyCaret was used to implement all the models. LGBM displayed the best model accuracy at 95.61%.

In [36] a feature selection model was used. In that model feature selection was implemented after the CTG dataset [19] was fed into a SMOTE module. SMOTE was utilized to resolve the imbalanced dataset. Their model was implemented on some of those models (already discussed in above mentioned other articles) that were relevant to the selected dataset. The best accuracy achieved was 95% using RF.

The authors in [37] implemented a neural network that was based on a Differential Privacy (DP) framework. In this model there were two binary classifiers. These classifiers jointly classified the CTG dataset [19]. Their proposed model displayed an accuracy of 91%.

In [38] a classification model based on an apriori algorithm was presented. That model was a combination of Adaboost, RF and feature selection. Moreover, the suspect class in CTG dataset [19] was merged into normal and pathological classes. This merger was performed to increase the overall accuracy of the proposed model.

Authors in [39], manually selected the suitable features of the CTG dataset [19] via Principal Component Analysis (PCA). At a later stage, those selected features were used as input to

an SVM module, that had been boosted with AdaBoost. That model achieved classification accuracy of 93%.

In [40], a self-learning algorithm was implemented. The model was in fact a Fuzzy C means clustering bases ANFIS model. The authors manually selected 9 features of the CTG dataset [19]. The model was able to achieve an accuracy of 96.39%.

The authors in [41], noted that for the CTG dataset [19] the two outputs have shared representations that allowed the proposed model to utilize shared features between the two outputs. This shared representation was useful in utilizing the two different outputs in this research thesis as well.

In [42] Cross-Industry Standard Process for Data Mining (CRISP-DM) was utilized for the first time for classifying CTG dataset [19] using different algorithms, the classification performance of which were compared. Explanatory Data Analysis (EDA) and Correlation Analysis (CA) was utilized by the author to determine the association between accelerations and decelerations of uterine contractions and also to observe the pattern of FHR histograms respectively. On Classification Analysis, it was observed that C5.0 outperformed other neural networks, achieving 90% accuracy for classification of fetal status.

For the CTG dataset [19], the performance comparison using multiple classifiers was implemented in [43]. The authors observed that performance depended on the size of the training samples and noise present in the dataset; a small error of a model did not always mean that the model was performing better; and K-Nearest Neighbors (KNN), which achieved the highest accuracy of 73.33%, classified better than Multi-Layer Perceptron (MLP).

Extreme Learning Machine (ELM), which operates like a single hidden layer feedforward neural network and has been conceived to address specific issues encountered faced in regression and classification, was used to classify CTG in the research [44], achieving 93.42% accuracy for classifying the patterns of FHR.

The above-mentioned literature review was for the Phase A and Phase B of this research work. The following literature review is for the Phase C of this research work.

In [45] a deep learning approach is proposed that used convolutional neural networks (CNNs) for anomaly detection in FECG signals. The authors achieved promising results, but their work did not specifically explore LSTM-based neural networks. This classifier aids technicians in identifying high- and low-quality segments during FECG recording. The model performance in determining signal quality was demonstrated by its 10-fold cross validated AUC (Area Under the Curve) of 0.95 after it was trained using FECG data obtained from 100 individuals. With this strategy, the automation of FECG recording and analysis might be improved, ultimately requiring less manual involvement.

FHR detection often calls for the detection of the fetal R-peak, which can be difficult, especially during times of low signal-to-noise ratio, such as those brought on by UC. In [46] a method for directly predicting the FHR from multichannel FECG using a convolutional neural network and a long short-term memory network was proposed. The network is trained using a dataset collected during labor, and the effectiveness of the technique is assessed on set-A of the 2013 Physionet/Computing in Cardiology Challenge as well as a test dataset.

In [47] the importance of FECG morphology in fetal health condition diagnosis is discussed. Due to contamination from maternal ECG (MECG) and other abdominal noise, extracting distinct FECG morphology is difficult. A two-stage Slow-Fast Long Short-Term Memory (SFLSTM) architecture proposed by the authors. While the fast LSTM eliminates residual noise, the slow LSTM filters out the MECG signals. Using public databases as a testbed, experiments reveal that this method increases the Signal-to-Noise Ratio (SNR) by 3.09 dB for MECG reduction and 1.81 dB for FECG improvement. The fast LSTM decreases computation by around 50% without sacrificing performance.

The problem statement regarding the accurate detection of fetal QRS is very important. The authors in [48] proposed a reliable Hilbert transform approach for FHR extraction from one or more data channels. This procedure can be automated and is unaffected by fetal position or movement. The Hilbert approach produces the greatest results, as compared to independent component analysis (ICA), ICA + Hilbert, and raw signals. In 46,789 fetal heartbeats, the analysis reveals around 171 errors.

Another issue regarding FECG signals relates to the FECG extraction from abdominal signal by removing the Maternal ECG signal. It can be difficult to extract FECG from non-invasive

sensors since it is weaker than maternal ECG (MECG) signals. Traditional approaches call for several sensors and complicated calculations. The research in [49] suggests a neural network-based method for distinguishing FECG data from abdominal ECG readings collected using non-invasive sensors in order to solve this issue. The proposed model successfully separated both the MECG and FECG signals within the maternal abdomen ECG signal using TensorFlow platform.

The authors in [50] proposed a unique method for detecting anomalies in FECG data using a multilayer long short-term memory (LSTM) autoencoder network. The network first utilizes LSTM to find patterns in the time series of the data, after which it reconstructs the errors and categorizes segments as normal or abnormal to find anomalies. The methodology uses a semi-supervised filtering algorithm to simulate ECG variability. The results displayed that proposed algorithm performs better than conventional ones and requires no previous knowledge. This method has the potential to improve the interpretation of fetal ECG data in common situations with precise signal identification.



3. RESEARCH DESIGN AND METHODOLOGICAL FRAMEWORK

3.1. Overview of Classification

In machine learning, classification is the grouping of entries into several categories according to a predetermined criterion. The categorization of data items into preset groups or categories based on their properties or qualities is a key idea in machine learning and data analysis. It is essential to several real-world applications, including fraud detection, picture identification, and email filtering.

Building a prediction model that can automatically classify new, unforeseen data points into one of many predetermined categories is the fundamental goal of classification. In this method, the model is trained using a labeled dataset in which each data point is assigned a specific class label. In order to create precise predictions about yet-to-be-observed data, the model then learns the patterns and relationships within the data.

There are several key components to understand in classification:

Features and Labels: Each data point in a classification issue is characterized by a collection of features, often known as characteristics or variables. The categorization model uses these characteristics as inputs. The model's intended outcome or objective for a particular collection of features is the class label.

Training and Testing: A training set and a testing (or validation) set are often separated into the dataset. The training set is used to teach the model how features and labels relate to one another. The model performance and generalizability to fresh, untested data are next assessed using the testing set.

Algorithms: There are several categorization algorithms, each having advantages and disadvantages. DT, RF, SVM, NB, and ANN are a few examples of typical algorithms. The selection of an algorithm is influenced by a variety of variables, including the complexity of the data, the number of characteristics, and the quantity of accessible data.

Evaluation Metrics: A number of assessment measures are used to gauge the effectiveness of a categorization model. The measures or metrics are accuracy, precision, recall, F1-score,

and ROC-AUC etc. These measures, which take into account things like true positives, true negatives, false positives, and false negatives, assist determine how effectively the model is doing across many classification-related elements.

Overfitting and Underfitting: It's important to strike the correct balance between a model that is overly complicated (overfitting) and one that is underly basic (underfitting). When a model learns noise from training data and performs badly on unobserved data, overfitting has taken place. On the other side, underfitting occurs when a model misses significant data patterns.

Feature Engineering: The selection, transformation, and creation of features from the raw data can have a big influence on how well a classification model performs. Proper feature engineering can increase the model capacity to retrieve pertinent data and its prediction accuracy.

3.1.1. K-nearest neighbors (KNN)

KNN predicts outcomes using the training data points that are most similar to a new input, working under the assumption that comparable data points have similar labels. By computing the separations between the points, it chooses the “k” closest neighbors and makes predictions using either a weighted average (for classification) or a majority vote (for regression). Its performance and success depend on variables like the number of neighbors and distance measures, while being straightforward and understandable. Its usage includes image identification, anomaly detection, and medical diagnosis, among other applications. For the best outcomes, factors like feature scaling and hyperparameter tweaking are also crucial to KNN. KNN is a nonparametric machine learning technique that is utilized for both classification and regression. The “k” closest training instances in a given feature matrix make up the input. For classification, the KNN output is based on a similarity measure between neighbors. Based on the value of k and the majority votes of the nearest members are classified as being part of a similar class. There are different distance functions to measure the distances among the neighbors. The “Minkowski” distance function formula is given in Eq. 3.1.

$$D(x, y) = \left(\sum_{i=1}^n |x_i - y_i|^p \right)^{1/p} \quad (3.1)$$

Where x and y are the points between which the distance is to be measured. Minkowski distance is a combined abstraction of both the Euclidean distance (if $p = 2$) as well as the Manhattan distance (if $p = 1$).

3.1.2. Multi-layer perceptron (MLP)

MLP is a type of feedforward neural network that can employ supervised learning through backpropagation to reduce errors in each iteration for classification applications. A MLP contains more than one hidden layer between one input layer and one output layer. Each neuron in the layers has an activation function that defines the output of that neuron when fed with any input. These linked layers of neurons are intended for tasks involving non-linear patterns. Each neuron takes in information, calculates biased weighted sums, and then applies activation functions. As an activation function, the rectified linear unit function (ReLU) is typically utilized. The function is defined as: $f(x) = \max(0, x)$. Weights of the links can be optimized (to decrease the error) iteratively using Stochastic Gradient Descent (SGD). Features are represented by input neurons, complicated interactions are captured by hidden layers, and outcomes are produced by the output layer. By iteratively modifying weights and biases, optimization methods are used to train MLPs to reduce loss functions. Despite issues like overfitting, MLPs are fundamental to deep learning and provide the basis for more complex architectures while also finding use in a variety of fields.

3.1.3. Support vector machine (SVM)

A flexible machine learning approach called a Support Vector Machine (SVM) is utilized for regression and classification. The gap between classes in data space is maximized by an ideal hyperplane that is calculated. SVMs use the “kernel trick” to translate data into higher dimensions, making them compatible with both linear and non-linear data. They employ a regularization parameter to balance accuracy and margin. SVMs use one-vs-one or one-vs-all strategies to solve jobs involving several classes. They perform well in high-dimensional settings, analyze a wide range of data, and generalize well. However, selecting kernels and parameters may be challenging, and SVMs may need a lot of computing power for huge

datasets. SVMs continue to be useful despite more recent approaches like deep learning, especially for moderate-sized datasets and easily interpretable models. SVM is a non-probabilistic classifier. It aims to map training points (from the sample) in space in such a way as to maximize the distance between different categories. Prediction is then performed on where the new points fall in that space. A special type of SVM called Support Vector Classification (SVC) is specifically used in classification. Different kernel function types can be used in SVC. One of the most popular ones is Radial Basis Function (RBF). The RBF kernel on two samples x_1 and x_2 (x_1 and x_2 have subsets) (that can be feature vectors in an input matrix) can be specified in Eq. 3.2.

$$K(x_1, x_2) = \exp\left(-\gamma * ||x_1 - x_2||^2\right) \quad (3.2)$$

3.1.4. Decision tree (DT)

It is a popular machine learning approach for classification and regression applications. It displays decisions and results in a tree-like structure. Nodes stand for decisions, branches for results, and the process begins at the root node. In accordance with measures like knowledge gain, the algorithm chooses qualities for judgments. Pruning and top-down induction are two methods to avoid overfitting. Decision trees can manage nonlinear connections, indicate feature importance, and are easily interpretable. However, they have the potential to be unstable, biased toward dominant major classes, and can overfit. By mixing decision trees, ensemble approaches like Random Forests (RF) overcome these problems. Essentially, decision trees provide a compromise between performance and interpretability in machine learning. DT is a tree-based decision tool. Each branch in the tree diagram represents a possible outcome. The root node represents the whole sample space. The decision nodes split the sample space based on a criterion. The leaf nodes are the endpoints that represent the classified elements. There are various methods that can be used as the splitting criterion: GINI index, Chi-Square, and Information Gain (via Entropy, which is the measure of impurity of the nodes). The formula for Entropy is given in Eq. 3.3.

$$E = - \sum_{i=1}^c p_i \log_2 p_i \quad (3.3)$$

Here the proportion of the samples that belong to the class c is denoted by p_i , for a particular node.

3.1.5. Random forest (RF)

Machine learning tasks including classification and regression employ the effective ensemble learning algorithm called Random Forest. Through bagging and random feature selection, it creates an ensemble of decision trees, each trained on a random sample of data and features. By combining the predictions of various trees through voting or averaging, this method maximizes accuracy while reducing overfitting. It is a popular option across different fields due to its adaptability, resilience with noisy data, and capacity to emphasize feature importance. Despite its advantages, RF has certain drawbacks. Although individual trees may be understood, it might be difficult to comprehend how the ensemble as a whole makes decisions. Hyperparameter optimization can also be very important for best performance. With a balance between accuracy, interpretability, and flexibility across many data settings, RF is a potent tool for developing precise and trustworthy prediction models. When the model is being trained, it can build many decision trees and combine them to provide a more accurate outcome. It addresses the problems caused by decision trees' overfitting. Splitting classes is decided using "Information Gain" at each node. There are several ways to gather this information. Entropy is one such method. The formula for entropy is already given in Eq. 3.3. Then, in Eq. 3.4 the formula for information gain is shown.

$$Gain(V, X) = E(V) - E(V, X) \quad (3.4)$$

Here, V is the target variable, X is the feature on which the data will be split, and Entropy $E(V, X)$ is the entropy determined following the split based on a feature X . The Gini index, often known as the impurity, is another method for determining the split criterion. This is a measurement of a collection of inequality (impurity) of different values. It achieves this by calculating the likelihood that a randomly selected case would be incorrectly labeled. This is then applied to evaluate the split quality. If c is the number of classes in the target variable and p_i is the percentage of values in the set that correspond to class i , then Eq. 3.5 displays the Gini impurity.

$$\text{Gini Impurity} = 1 - \sum_{i=1}^c (p_i)^2 \quad (3.5)$$

3.1.6. Feature importance

A key idea in several disciplines, including machine learning, statistics, and data analysis, is feature importance. It describes a way to quantify how much a certain characteristic or variable affects the functionality or result of a model, algorithm, or study. For making educated judgments, figuring out what causes a phenomenon, and improving the performance of prediction models, it is essential to comprehend feature significance.

Feature significance aids in determining which input variables (features) have the most influence on the predictions made by the model when used to machine learning and predictive modeling.

The features are ranked based on their importance for the following reasons:

The importance of a feature sheds light on the prediction process of a given model. It aids in comprehending the connection between input variables and model output.

Usually not all traits and features are equally significant. Some could contain information that is redundant or useless, which might cause overfitting or increase computing complexity. The process of feature selection is guided by feature importance ratings, and less significant elements can be removed to streamline the model and boost performance.

The underlying issue of a dataset can be evaluated by observing the relevance of the features. The relevance can be found by checking the individual importance of all the features of that dataset.

Feature importance can aid in the detection of potential biases in models. If some features are given abnormally high weights, it may be a symptom of unfair or biased forecasts based on delicate characteristics like gender or race.

Various methods to find out about the feature importance are given as follows:

DT, RF and Gradient Boosting (GB) are tree-based algorithms that build decision trees and assess the contribution of each feature to the decrease of prediction impurity. The significance score of a feature is determined by the feature's overall contribution to all trees.

Another approach uses permutation, in such a manner that the values of a single feature are shuffled at random, and the reduction in model performance is measured. Higher relevance is indicated by a larger decrease.

For linear models, the magnitudes of the coefficients are frequently used to estimate the significance of the features. Higher relevance is often indicated by larger coefficients.

Features having non-zero coefficients after regularization are considered as important in LASSO (Least Absolute Shrinkage and Selection Operator) regression.

Shapley Additive explanations (SHAP) (discussed separately later in this thesis) values offer an explanation method of any machine learning model output. Each feature is given a value that corresponds to how much it contributed to the prediction.

Methods like sensitivity analysis or gradient-based methods may be used to gauge the importance of features in deep learning and neural network models.

It is significant to know which type of strategy is used in which type of dataset. Additionally, as correlation does not necessarily indicate causation, feature importance should be taken with caution. Features may seem significant because of ambiguous connections or confounding variables. The performance of models in many domains may be understood, interpreted, and improved with the use of feature importance. It gives the ability to take well-informed judgments, glean valuable information from data, and make sure that models are legitimate, just, and efficient.

The mathematical formulae to supplement feature importance calculations are as follows:

The decrease in the node impurity that is weighted by the probability of reaching that node is termed as the importance of a feature. Where the number of samples that reach the node, divided by the total number of samples is called as the node probability. The importance of a feature is directly proportional to the value obtained in the above-mentioned process. Thus,

the higher the value the more important the feature. The algorithm of calculating the importance of a node using “GINI Importance” is given in Eq. 3.6.

$$Nimp_j = W_j C_j - LeftW_j LeftC_j - RightW_j RightC_j \quad (3.6)$$

here $Nimp_j$ is the importance of node j , W_j is the weighted number of samples reaching node j , C_j is the impurity value of node j , $Left_j$ is the child node from the left split at the node j , $Right_j$ is the child node from the right split at the node j . Then the importance of feature is given in Eq. 3.7.

$$Fimp_i = \frac{\sum_{j=NS} Nimp_j}{\sum_{k \in AN} Nimp_k} \quad (3.7)$$

here $Fimp_i$ is the importance of feature i , $Nimp_j$ is the importance of node j , NS is the node j that splits on feature i , and AN is the total nodes. The importance of features can be normalized (between 0 and 1) by dividing by the sum of all feature importance values within that tree as given in Eq. 3.8.

$$Normfimp_i = \frac{fimp_i}{\sum_{j \in AF} fimp_j} \quad (3.8)$$

here AF is all the features. Finally at Random Forest level, the final importance of feature will be the average over all the decision trees, is given in Eq. 3.9.

$$RFimp_i = \frac{\sum_{j \in AT} Normfimp_{ij}}{T} \quad (3.9)$$

here $RFimp_i$ is the importance of feature i that is calculated from all the trees in the Random Forest model, $Normfimp_{ij}$ is the normalized feature importance for i in tree j , AT is all the trees, and T is the total number of trees.

3.1.7. Long short-term memory (LSTM)

Long Short-Term Memory (LSTM) is a deep learning model. It is a form of recurrent neural network (RNN). LSTM is generally suited to capture long-term dependencies and contextual information. It is well-suited for sequence data analysis and has been successfully applied in

various domains. LSTM incorporates memory cells that can store and retrieve information over extended time intervals. This memory mechanism enables LSTM to retain important information while discarding irrelevant details. LSTM's ability to learn from and process sequential data makes it ideal for time series analysis tasks, such as predicting and detecting abnormalities in fetal electrocardiogram (FECG) signals.

The main components of LSTM are as follows:

The cell state is LSTM main memory, and it goes along the whole sequence. Using a variety of procedures, data may be added to or withdrawn from the cell state, enabling the LSTM to recall or forget data as necessary.

The input gate regulates how much of the fresh data should be incorporated into the cell state. It has a sigmoid layer that selects which values should be updated.

The forget gate decides what information from the previous cell state should be forgotten. For each element in the cell state, it outputs a value between 0 and 1, signaling whether to keep or forget the information, using a sigmoid layer.

The output gate controls the data that is delivered to the LSTM hidden state and output. Based on the input received at the moment and the prior concealed state, it determines which portions of the cell state should be disclosed.

The hidden state (or also called output) is LSTM cell output, which sends data to the following time step. It is calculated using the cell state following modifications performed on the input, forget, and output gates.

The LSTM architecture has the capacity to learn long-range relationships. This makes it well-suited for a variety of sequential data processing applications, including time series forecasting, speech recognition, and natural language processing. Due to its capacity to handle both short-term and long-term patterns in data, LSTMs have proved essential in producing state-of-the-art outcomes in several applications. There are many different flavors of LSTM. However, in this research study only the classic LSTM is used. The reason for that choice is that it is robust and stable. Overall, LSTM is a strong RNN architecture that successfully solves the vanishing gradient issue, enabling it to capture complex relationships

in sequential data. With the help of its memory cells and gating mechanisms, deep learning for sequential tasks has significantly advanced in a number of different fields.

3.2. Feature Extraction

In the domains of signal processing, computer vision, natural language processing, and machine learning, feature extraction is a key idea. It entails converting raw data (which may be complicated and high-dimensional) into simple features, a more concise and useful representation. These features, which are intended to make additional analysis, categorization, or other activities easier, capture the key aspects of the data. Raw data may be immensely varied in numerous applications, ranging from photos and audio signals to written documents and sensor measurements. Raw data frequently exhibits patterns that are difficult to immediately identify, is noisy, contains useless information, or both. The most important features of the data are found and represented using feature extraction in an effort to reduce these problems.

All methods that enable the creation of new features from an existing dataset are collectively referred to as feature extraction (FE). The original dataset might then be obtained by using these additional characteristics once more.

A brief overview of the process of feature extraction is as follows:

Data preprocessing: In order to eliminate noise, fix mistakes, and standardize the format of the data before feature extraction, preprocessing is frequently required. This process guarantees that the data is reliable and consistent.

Dimensionality Reduction: The issue of working with a high dimensional data makes it difficult to work in practical scenarios. Overfitting and higher computing complexity may result from it. To minimize the number of features while maintaining as much relevant information as feasible, feature extraction frequently uses dimensionality reduction techniques like Principal Component Analysis (PCA) etc.

Transformations: Various transformations may be used, depending on the type of data. Features in image processing might be edges, textures, or color histograms. Word

frequencies, TF-IDF (Term Frequency (TF) - Inverse Document Frequency (IDF)) scores, or even more sophisticated embeddings like BERT are examples of features in text analysis.

Compatibility with Machine Learning: Extracted features should be created with the final objective in mind. If categorization is the aim, the characteristics must effectively discriminate across classes. The characteristics should group related instances together if clustering is taking place.

Domain information: The efficacy of feature extraction may be considerably improved by including domain-specific information. Engineers and researchers frequently have insights about which data elements are essential for resolving a certain issue.

Iterative Process: The process of extracting features is frequently iterative. Engineers could experiment with various transformations and methodologies, gauge how they affect the performance of the model, and then modify their strategy in light of the findings.

Evaluation and Validation: The extracted characteristics must be verified and assessed in light of the job at hand. In order to do this, machine learning models might be trained using the extracted features, and their performance could be evaluated using measures such as accuracy, precision, recall, F1-score, etc.

In summarized terms, feature extraction is a critical step in simplifying and making sense of complicated data. As a result, analysis, modeling, and decision-making are made possible in a variety of applications more quickly and effectively. It converts raw data into a representation that captures pertinent patterns and attributes.

3.2.1. Autoencoder

Autoencoder (AE), which is straightforward and adaptable, is a deep learning method to implement feature extraction. The basic concept of an autoencoder is to learn a condensed and compressed representation of the input data, which is sometimes referred to as the “encoding” and then to “decode” this representation back into an approximate representation of the original data. By removing noise or irrelevant information, this approach aids in identifying the most significant characteristics or patterns in the data. Data compression, denoising, and feature extraction are some of the uses for autoencoders.

An autoencoder attempts to reproduce the original input after first attempting to understand the relationships between the characteristics of an input data.

An autoencoder should be able to balance between the following two qualities:

- For the input, it should be sensitive enough to accurately build a reconstruction.
- For the input, it should be insensitive enough to not simply memorize (overfit) the input data.

A normal autoencoder is limited by these different qualities. This limits the autoencoder to learn only those input trends or patterns which later helps it in recreating the input (like the original). At the same time this balanced encourages the autoencoder to ignore the repeated and extra useless entries in the input. A loss function (as displayed in Eq. 3.10) has a regulator that prevents the autoencoder from memorizing all of the input and a reconstruction loss (RL) that forces the autoencoder to be responsive to input.

$$LossFunction = RL(x, x^o) + Regulator \quad (3.10)$$

There are many different flavors of autoencoders, like:

Sparse Autoencoder: To encourage the learnt representation to be sparse in this autoencoder, an extra regularization term is added to the loss function. As a result, only a tiny portion of the neurons in the hidden layers are triggered for a given input.

Denoising Autoencoder: It is made to pick up reliable features by practicing on damaged input data samples. The network is anticipated to produce clean reconstructions even when given noisy or partially damaged input during training.

Variational Autoencoder (VAE): This autoencoder design has a probabilistic bent. It discovers how to create a probabilistic distribution in the latent space, giving it greater freedom to produce fresh data samples.

Convolutional Autoencoder: Convolutional neural networks are used in the construction of this autoencoder, which makes it especially well-suited for tasks requiring pictures or spatial data.

Applications for autoencoders may be used in a variety of fields, including anomaly detection, dimensionality reduction for visualization, image denoising, and even generative modeling. They may be considered the foundation for deeper learning architectures and more complicated models like generative adversarial networks (GANs).

Autoencoders are a useful tool for extracting key characteristics from data and lowering its dimensionality in the field of machine learning. Autoencoders contribute to a variety of tasks that call for effective data compression, denoising, and feature extraction by learning meaningful representations.

Basic architecture of autoencoder is as follows:

Encoder: The encoder converts input data into a representation with fewer dimensions. In comparison to the original data, this lower-dimensional representation often comprises fewer dimensions or characteristics, which encourages the network to take in the most important information. It can be made from convolutional or MLP modules.

Decoder: The decoder uses the encoder's lower-dimensional representation to try to piece together the original input data. A usable and compact representation that can be utilized for reconstruction must be learned by the network in order to provide an output that is as similar to the original input as feasible. Similar to encoder, it can also be made from convolutional or MLP modules.

Bottleneck: The encoder compresses the input data so that it may be sent to a bottleneck, this compression is done in a step-by-step manner. To compel the neural network to remember just the trends in the input entries rather than the complete input data is the main objective of bottleneck. The decoder then uncompresses the bottleneck and tries to bring the data back into its original state.

3.3. Feature Selection

By removing the least significant or redundant characteristics from a dataset, feature selection (FS) can lower the dimension of the dataset and, in certain situations, boost the model efficiency. Feature selection is quite helpful in situations where a huge dataset is using a lot of system memory.

The goal of feature selection, a crucial step in machine learning and data analysis, is to find and choose the most pertinent and instructive characteristics of a given dataset. In pragmatic terms, it involves selecting the variables or features that will have the most impact on the model performance and avoiding those that are unnecessary or redundant. Effective feature selection may increase model accuracy, decrease overfitting, speed up training, and improve the ability of the model to be understood easily.

Because not all characteristics in a dataset are equally significant or valuable, feature selection is required. Subpar model performance may result from features that induce multicollinearity, carry noise, or have low predictive power. Furthermore, adding too many features might result in the “curse of dimensionality”, making the model more complicated and challenging to train while also raising the danger of overfitting.

There are many methods with which feature selection can be implemented, as shown as follows:

Filter approaches: These approaches use statistical measures to rank and score characteristics according to how relevant they are to the target variable on an individual basis. Correlation, mutual information, the chi-squared test, and variance thresholds are examples of common metrics. Then, depending on their scores (often with a specified threshold) features are chosen.

Wrapper techniques: To assess certain feature subsets, wrapper techniques employ machine learning algorithms. This method requires repeatedly training and assessing the model on various feature subsets, which might be computationally demanding. This category includes methods like forward/backward selection and Recursive Feature Elimination (RFE). RFE is discussed in detail later in this section.

Embedded techniques: During the model training phase, embedded techniques do feature selection. By penalizing or omitting specific features during training, some algorithms, such as LASSO (Least Absolute Shrinkage and Selection Operator) and decision trees, automatically involve feature selection.

Hybrid techniques: To increase the efficacy and efficiency of feature selection, hybrid techniques incorporate components of other methodologies. For example, combining the filter and wrapper methods can benefit from the strengths of both approaches.

The kind of the dataset, the computing resources available, the particular machine learning method being used, and the intended trade-off between model complexity and interpretability should all be taken into account when choosing a feature selection technique.

It is critical to remember that feature selection does not offer a universally applicable answer. It is also possible that what works well for one dataset or situation would not be best for another. Additionally, the effect of feature selection on model performance is influenced by the quality of original feature set and variety as well as the machine learning algorithm of choice.

The performance and interpretability of models may be greatly impacted by feature selection, which is an essential stage in the machine learning process. Practitioners may build more effective and accurate prediction models while getting insights into the underlying trends in the data by carefully selecting key characteristics and removing extraneous ones.

3.3.1. Recursive feature elimination

Recursive Feature Elimination (RFE) is a machine learning method that is used to choose features, which is a crucial stage in the creation of predictive models. RFE is a wrapper-type supervised method. Datasets can contain a lot of features (also called variables or characteristics) in many real-world applications, some of which might not be very important to the performance of the model. In fact, adding pointless or duplicate features might increase overfitting, increase computing complexity, and reduce the capacity to understand the model. RFE resolves this problem by methodically deleting the least significant features of a given dataset, repeatedly enhancing the feature set of the model until a specified stopping condition is satisfied. The main principle of RFE is to train the model and assess its performance after eliminating one or more features each iteration. RFE assists in identifying and keeping the most relevant features while eliminating those that bring little to no extra value by analyzing the influence on the model performance. The reason for selecting RFE method to implement feature selection is due to its flexible nature.

A simple version of the working of the RFE method is given as follows:

- Develop the classifier by training it.
- Determine the rating criteria for every feature. A rank list is made.
- Exclude the feature that has the lowest ranking criterion, then repeat the process until a stopping requirement is met.

A detailed version of the working of RFE method is given as follows:

1. Start by using all of the dataset attributes to train a machine learning model (often a supervised learning technique). The first model used will depend on the issue at hand and may include neural networks, support vector machines, random forests, or linear regression.
2. Features are prioritized according to their relevance ratings once the first model has been trained. The model coefficients for linear models, feature importance for tree-based models (like DT and RF), and other approaches like permutation importance may all be used to get the importance scores.
3. The dataset is purged of the feature with the lowest relevance score. After then, the model is retrained using the smaller feature set.
4. Depending on the kind of issue (classification or regression), the model performance is assessed using a validation metric, such as accuracy, F1-score, or mean squared error.
5. The procedure continues by either deleting a fixed number of features at each iteration or until a desired number of features is attained, depending on the stopping criterion used.
6. The steps from 3 to 5 are performed repeatedly, omitting the least significant aspect after each iteration. Each cycle ends with a performance evaluation of the model.
7. The process continues up until a predetermined terminating condition is satisfied, such as when all needed characteristics are obtained, or the model performance starts to deteriorate.
8. The remaining features are deemed to be the most pertinent for the specified problem once the stopping condition has been met. The final model is then trained using only these chosen characteristics, producing a model that is more condensed, effective, and maybe interpretable.

RFE helps to concentrate on the most informative features, striking a balance between model complexity and performance. In situations when the feature space is highly dimensional, it can aid in enhancing generalization, lowering overfitting, and improving model interpretability.

3.4. Overview of Hyperparameter Optimization

Hyperparameter optimization is a very important element in deep learning. Contrary to the standard parameters of the model, which are changed during training, hyperparameters are values that are not directly learnt from the training data. Instead, the capacity, complexity, and behavior of the model are affected by the training process's numerous parameters, known as hyperparameters.

Finding the ideal hyperparameter combination that gives a machine learning model the highest performance is the goal of hyperparameter optimization. The objective is to achieve a compromise between underfitting (where the model is too simplistic to capture patterns in the data) and overfitting (where the model matches the training data too closely but performs badly on fresh data).

A step-by-step explanation of the hyperparameter optimization process is given as follows:

Determine and define which hyperparameters should be tuned for a particular model and job. These could include dropout rate, regularization strength, batch size, number of hidden layers, number of neurons in each layer, and learning rate etc.

Select an optimization approach and strategy to find the ideal set of hyperparameters. Grid search, random search, and more sophisticated techniques like Bayesian optimization or evolutionary algorithms are examples of common tactics.

Define a range of values or a distribution for each hyperparameter that the optimization algorithm will consider. This range of values is also called search space in hyperparameter optimization. For instance, the range (0.001, 0.1) may be searched for the learning rate, while the set (1, 2, 3) might be used to select the number of hidden layers.

Select an evaluation metric to assess the model performance. These metrics might include recall, F1-score, accuracy, precision, mean squared error etc. This measure is used by the optimization algorithm to evaluate the performance of various hyperparameter combinations.

During the search process the optimization method examines the specified search space iteratively by experimenting with various combinations of the hyperparameters. The model is tested on a different subset of data (validation set) after being trained on a subset of the data (training set) for each combination. Calculated evaluation metrics are used to determine whether or not the algorithm should change its search direction.

Early pausing and stopping is frequently used in search operations (or processes) to avoid overtraining and possible overfitting. This refers to halting the training of a model if its performance on the validation set begins to deteriorate.

Final evaluation is made based on the performance metric results, the optimal set of hyperparameters is chosen after the optimization process. The final model is then trained on the complete training dataset using this optimum collection of hyperparameters.

To gauge the finished model's capacity to generalize to new data, its performance is assessed using a different testing dataset.

The “tuning of a problem” analogy applies to hyperparameter optimization. This is accomplished by manipulating specific parameters (that may be used to influence the learning process) in order to identify the values that can solve the machine learning issue in the best possible way. A preset loss must be minimized, or the precision of an objective function must be maximized given independent data. When given a tuple of hyperparameters, the "objective function" returns the corresponding loss. Cross-validation is most frequently employed to calculate this generalization performance.

The commonly used search algorithms used for performing hyperparameter optimization are as follows:

The simplest way to carry out hyperparameter optimization is by using Grid Search. It just conducts a thorough search on hyperparameters that the user specifies. The users must have

a basic understanding of the hyperparameters in order to get the most of this method. If not, either the search time will lengthen or significant hyperparameter values will be disregarded. Additionally, a limited search space is optimal for grid search. The time it takes to find a decent collection of hyperparameters will expand exponentially if the search space is big [51].

Random search might be viewed as an improvement over Grid search. Over the hyperparameter search space, it does a random search. The search can be stopped by letting the operation exhaust the predetermined budget or until the requisite precision is reached. Random search offers several benefits even though it is similar to grid search. For instance, random search may perform better than grid search in a situation where the hyperparameters are not distributed equally. The likelihood of discovering the ideal hyperparameters increases as the time allotted for random search increases. This approach is also called the Monte Carlo approach [52], which has become widely used when dealing with massive multidimensional datasets in deep learning [51].

The Bayesian method is the foundation of Bayesian optimization (BO) [53]. The global optimum is sought using this sequential approach, which requires the fewest repetitions. Its main objective is to balance exploitation and exploration, which is the process of coming to the optimal conclusion based on the facts at hand. It is a model that seeks to assemble additional data [51]. When using BO, every issue is seen as a mystery or a black box. An objective probability surrogate model is developed. The best hyperparameter values are then chosen and every effort is made to arrive at the optimal solution based on the outcomes of prior rounds.

The following are the steps in the Bayesian optimization algorithm:

1. The prior distribution of a surrogate model is created.
2. The surrogate model's top-performing hyperparameter set is found.
3. The computation of the acquisition function for the present surrogate model.
4. The goal function is used with the chosen set of hyperparameters.
5. The new data is added to the surrogate model.

Until the resource is depleted or a suitable collection of hyperparameters is identified, the above-mentioned iterations are repeated.

3.5. Synthetic Minority Oversampling Technique

Machine learning is frequently hampered by imbalanced datasets, when one type of data is vastly underrepresented in comparison to another. As a result of the minority class's sparse representation in the data, the algorithm may find it difficult to learn it, which might result in biased model performance. This problem is addressed and the performance of machine learning models on unbalanced datasets is improved by Synthetic Minority Oversampling Technique (SMOTE), a widely used technique.

SMOTE creates artificial instances for the dataset's minority class as part of its operation. In order to do this, it chooses one instance from the minority class and locates its k closest neighbors. Then, along the line segments connecting the selected instance with its neighbors in the feature space, it generates additional synthetic instances. By successfully producing additional cases that fall into the minority class's distribution, this method increases the minority class's representation in the dataset.

When the distribution of the classes in the desired result is equal, the dataset is said to be balanced. An unbalanced dataset, on the other hand, has an uneven distribution of classes. When classifying an uneven dataset, the results may be skewed so that the model receives less data pieces (entries) for minority classes. As a result, the model's performance for minority classes may decline. Methods like under sampling (the majority class) and oversampling (the minority class) can be used to balance out an unbalanced dataset. Under sampling can be simpler to implement since it randomly removes entries from the dataset's majority class, but it is not ideal, especially in a dataset with a strong medical focus where each entry is significant. On the other hand, oversampling, in which the minority class is raised to match the majority class, is a preferable alternative. The minority classes can be duplicated at random numerous times to create random oversampling. By interpolating minority class entries into the feature space of the data, a different oversampling method known as SMOTE creates synthetic entries in the dataset. By establishing new instances between a minority sample and its k neighbors, SMOTE accomplishes this. The characteristics of the new entries aren't exact replicas of those of the old entries; instead, they

draw inspiration from their close neighbors. An unbalanced dataset's minority class distribution is the only one that SMOTE impacts. The class in the majority is unaffected.

The benefits of using SMOTE are as follows:

- It allows to balance class distribution that in turn allows the dataset to accurately reflect the trends in it.
- It allows dataset to become balanced by reducing bias toward the majority class.
- It allows the feature space to increase thus, causing the models to recognize more intricate patterns in the new dataset.

It is a potent method, but there are a few things to keep in mind while implementing it, which are given as follows:

The selection of k (which is the number of nearest neighbors, or the parameter k) is very important as it can affect how well synthetic samples turn out. While a greater k could result in oversampling from unrelated regions, a smaller k might provide too localized synthesis.

Producing an excessive number of synthetic instances may cause the synthetic data to get overfitted. Finding the right balance between producing enough samples and avoiding overstuffing the dataset with artificial data is crucial.

To further improve model performance, SMOTE may be used in conjunction with other approaches like ensemble methods.

For correcting class imbalance in machine learning, SMOTE is an effective method. It enables models to learn and generalize from unbalanced datasets better by synthetically boosting the minority class, thereby enhancing the accuracy of forecasts and judgments.

Based on the original research work regarding SMOTE [54], a simple version of SMOTE algorithm is given as follows:

1. A minority class instance is selected at random.
2. The k closest neighbors, for the selected instance, are selected from the minority and majority classes of the selected instance.

3. A synthetic instance for each of the closest neighbors by interpolating between the candidate instance and the neighbor is produced. A random number between 0 and 1 determines how much interpolation will occur.
4. Steps 1 to 3 are repeated as many times as necessary for creating artificial points (entries in the dataset).

3.6. Overview of Performance Analysis

For models proposed in Phase A and Phase B, the performance analyses were judged based on metrics for precision, F1-score, accuracy, specificity, and recall. If TP represents the true positives, TN represents the true negatives, FP represents the false positives, and FN represents the false negatives then formulae of the performance metrics are given in Table 3.1. Utilizing 5-fold stratified cross-validation, the Phase A and Phase B models were tested. Because the CTG dataset [19] was not balanced, stratified cross-validation was used. Although the imbalance was corrected using SMOTE, care was still taken to prevent the dataset's innate imbalance. Additional performance metrics that were used to evaluate the models were the precision-recall curve, area under the ROC curve (AUC), and receiver operating characteristic (ROC). The ROC and AUC performance measures were utilized in this study because they may be used to show readers how feasible it is to adopt the suggested model and to persuade them to do the same in their own research. The use of ROC and AUC in this study was also motivated by the need to offer an intuitive explanation of the outcomes of the suggested mode. As a result, even readers with no prior knowledge of ML (machine learning) may clearly comprehend how well the proposed models perform. Medical professionals might also comprehend the performance of the proposed models because the topics covered by this research study also encompass the medical domain. The SHAP (SHapley Additive exPlanations) analysis (explained later in this section) was also carried out to further improve the interpretability of the suggested model. Matthew's correlation coefficient (MCC) (explained later in this section) was only used in the performance analysis as well as in the proposed algorithm of Phase B.

Table 3.1. Metrics Used for Measuring the Performance

Metrics	Metric Formula
Accuracy	$\frac{TP + TN}{TP + TN + FP + FN}$
Precision	$\frac{TP}{TP + FP}$
F1-Score	$\frac{2TP}{2TP + FP + FN}$
Recall (Sensitivity)	$\frac{TP}{TP + FN}$
Specificity	$\frac{TN}{TN + FP}$
MCC	$\frac{(TP * TN) - (FP * FN)}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}}$

3.6.1. Confusion matrix

A crucial tool in machine learning and statistics for evaluating the effectiveness of a classification model is the confusion matrix. By dividing the outcomes into many categories, it offers an extensive approach to illustrate the precision and efficacy of a model's predictions. With binary or multiclass classification issues, the confusion matrix is very helpful. The following labels are used while discussing about confusion matrix:

True Positives (TP) are situations where the model properly anticipated a positive outcome, but the situation really turned out to be positive. To put it another way, the model accurately recognizes instances that belong to the positive class.

False Positives (FP) are situations that the model mistakenly predicts as positive but are actually negative. This is a type of error while classifying items in the input, in which the model incorrectly assigns instances to the positive class.

True Negatives (TN) are situations where the outcome is genuinely negative, and the model properly anticipated that it would be. Instances that don't fit within the positive class are correctly identified by the model.

False Negatives (FN) are situations that the model mistakenly forecasted as negative but were really positive. This is also a type of error while classifying items in the input, in which instances that belong to the positive class are not recognized by the model.

For binary classification cases, a confusion matrix consists of four blocks. In this research study, for Phase A and Phase B since the classification is multivariate thus the confusion matrix would have more than four blocks. Other performance metrics depicted in Table 3.1, also are derived from the values found in the confusion matrix. Brief overview of those performance metrics are as follows:

Accuracy is the correct prediction ratio of a model.

Precision is the percentage of accurately predicted positive cases out of all instances projected as positive. It gauges how well the model can steer clear of false positives.

Recall (also called sensitivity or true positive rate) is the percentage of properly anticipated positive cases out of all actual positive instances. It assesses the model's capacity to locate each pertinent positive case.

F1-Score is a harmonic mean of accuracy and recall. It offers a fair assessment of a model's effectiveness.

Specificity (also called true negative rate) is the percentage of accurately predicted negative cases out of all actual negative instances. It gauges how well the model can steer clear of false negatives.

Strengths and limitations of a model in categorizing various scenarios can be better understood by examining these performance metrics and the confusion matrix. Making educated judgments regarding the model deployment, suitable assessment criteria selection, and model fine-tuning all depend on this data.

3.6.2. Receiver operating characteristic curve

One of the most important fundamental ideas in statistical analysis and machine learning is called Receiver Operating Characteristic (ROC). It is frequently employed to assess the effectiveness of binary classification models. Moreover, after certain modification in the code, ROC can also be used for multivariate dataset, by taking up one class at a time. The True Positive Rate (TPR), often referred to as sensitivity or recall, and the False Positive Rate (FPR) are trade-offs that are graphically represented by the ROC curve. The ratio of

properly anticipated positive cases to all real positive instances is known as the true positive ratio (TPR), whereas the ratio of erroneously predicted positive instances to all actual negative instances is known as the false positive ratio (FPR). Plotting these two rates at different categorization threshold values, from 0 to 1, results in the ROC curve. A quantitative measurement obtained from the ROC curve is called the AUC, or Area Under the ROC Curve. It provides a single number that encapsulates the entire effectiveness of a binary classification model. An AUC of 0.5 represents random performance, whereas an AUC of 1 denotes excellent discrimination between the two groups. The AUC scales from 0 to 1. Better model performance in differentiating between positive and negative examples is shown by higher AUC values. The AUC values are understood as follows:

- Coin-flipping-like random performance for AUC value of 0.5.
- Poor to fair discrimination model capacity for AUC values of 0.5 to 0.7.
- Reasonable discrimination model capacity for AUC values of 0.7 to 0.8.
- Good discrimination model capacity for AUC values of 0.8 to 0.9.
- Outstanding discrimination model capacity for AUC values of 0.9 to 1.

Benefits of using ROC and AUC are as follows:

As AUC emphasizes the ordering of predictions rather than their absolute values, it is resilient to class imbalance. As a result, it might be a helpful statistic for analyzing datasets that are unbalanced. The model performance is taken into account across all feasible threshold values, giving an overview of the general capacity of the model to distinguish classes without being influenced by a particular threshold selection. AUC makes it simple to compare various models. Better model performance is typically correlated with higher AUC values. AUC is useful in determining how consistently accurate predictions made by a model are under various threshold conditions. In circumstances involving unbalanced classes or when comparing multiple models, ROC and AUC offer a thorough perspective of the model performance.

3.6.3. Precision recall curve

When working with imbalanced datasets, a classification model performance is assessed using a graph called a Precision-Recall (PR) curve. In such a graph recall is plotted on the

x-axis and precision is plotted on the y-axis results. It sheds light on the ability of the model to accurately identify positive occurrences (high precision) and capture all accessible positive examples (high recall). When the distribution of the classes is skewed, or when one class has much more instances than the other, the PR curve is very helpful. In these circumstances, accuracy alone may be deceptive as a performance metric since a model that consistently predicts the majority class may still have high accuracy even if it is unable to identify the minority class. A PR curve is a useful method for assessing the effectiveness of classification models, particularly when there are class imbalances. It provides perceptions into how effectively a model balances the trade-offs between precision and recall, assisting practitioners in making decisions based on their own needs and goals. Since the original CTG dataset used in this research study (Phase A and Phase B) was unbalanced; Thus, PR curve was implemented to see the actual performance of the proposed models. PR curve advances from the point (0, 0) towards the top-right corner. The perfect point on the PR curve is (1, 1), which stands for flawless recall and precision values. In actuality, though, this stage is frequently unattainable. A PR curve that is more inclined to be in the top-right corner indicates a strong model. The area under the PR curve is frequently used as a summary statistic to assess the performance of models. Better performance is indicated by a greater area under the PR curve.

3.6.4. SHapley additive exPlanations (SHAP) analysis

SHAP analysis is a potent method used in data science and machine learning to explain the predictions generated by sophisticated models. It offers a means of comprehending and interpreting the contributions of specific traits, or features, or variables to the forecasts or predictions made by a model. The Shapley value, which is used to equally share the value of a coalition among its members, is a term that SHAP analysis, which has its roots in game theory, derives from cooperative game theory. The main objective of SHAP analysis is to deconstruct a model's forecast for a given instance into the contributions of each feature, identifying the sources of variation between the model's prediction and a baseline prediction. The basic working principle of SHAP is given as follows:

A baseline prediction is made first, followed by a comparison. This may be a reference point that makes sense in the context of the situation or a baseline average forecast. The goal is to

evaluate how the forecast or prediction made by the model for a given instance differs from the baseline prediction.

SHAP values are determined by taking into account all potential characteristics and their contributions. By comparing the model prediction, when a particular feature is included vs when it is omitted, while holding other features constant, one may determine a feature's contribution for a given feature. The size of the prediction gap shows how much that particular characteristic affected the result.

SHAP analysis distributes the overall influence of all aspects fairly among the individual features by using the Shapley values notion from cooperative game theory. According to Shapley values, each feature must make a meaningful contribution that is consistent with how it interacts with other features.

The consistency and additivity properties of the SHAP analysis are among its main advantages. The model's prediction for a given instance is the product of the SHAP values for each feature added to the baseline prediction. This makes sure that the explanations offered by SHAP values match the overall forecast of the model.

A number of approaches, including waterfall plots, force plots, and SHAP summary plots, may be used to view SHAP values. These visuals aid analysts and stakeholders in comprehending the relative contributions of various variables and how they work together to produce a certain forecast.

Due to its capacity to offer clear and understandable explanations for intricate models, including machine learning techniques like random forests, gradient boosting, neural networks, and more, SHAP analysis has become increasingly popular. It may be applied to a variety of tasks, such as financial modeling, medical diagnosis, picture classification, and credit scoring. A sophisticated method for comprehending the variables that affect model predictions is SHAP analysis. By decomposing predictions into feature contributions, it improves the interpretability, transparency, and accountability of machine learning models, making them more reliable and valuable for making decisions. Using SHAP analysis, one can describe how machine learning models worked. It seeks to show how various characteristics or input items affect a model's capacity for prediction. The main goal of

SHAP analysis is to calculate the Shapley values, which are based on cooperative game theory [55]. These numbers can be used to express how much each characteristic contributed to the outcome. By averaging across all potential feature permutations, these figures give an approximate indication of the marginal contribution of each feature to the forecast. The following advantages can result from applying SHAP analysis in machine learning:

1. Explain the importance of the input characteristics.
2. Modify the weights to change how much the characteristics affect the model.
3. To identify prejudice or bias.
4. To make the model easier to comprehend.

3.6.5. Matthew's correlation coefficient

An indicator frequently used to assess the effectiveness of classification models is the Matthew's Correlation Coefficient (MCC), particularly when there are unbalanced class distributions in a given dataset. Compared to typical accuracy measurements, it offers a more thorough assessment, particularly when analyzing datasets when one class dominates the other. When there is an imbalance between the classes (when one class is much more frequent than the other) MCC is particularly helpful. In these circumstances, only using the accuracy metric might be deceiving since a model that consistently predicts the majority class would nevertheless have a high level of accuracy. MCC, on the other hand, takes into account all four potential outcomes of a classification job and offers a more complex view of a model performance. MCC is a reliable statistic for assessing the effectiveness of binary classification models, particularly in situations when the datasets are unbalanced. While accounting for false positives and false negatives, it offers a fair evaluation of genuine positive and true negative rates. As a result, it serves as a useful tool for evaluating model performance in a range of applications, such as medical diagnosis, fraud detection, and anomaly identification. MCC performance metric is used for Phase B only. Moreover, MCC is also used as an important module in the proposed algorithm for Phase B. The usage of MCC for Phase B is discussed later in section 3.8 and in the discussions section. The primary reason to use MCC in Phase B is due to the fact that the class distribution in selected CTG dataset [19] is imbalanced (as observed from Figure 1), thus, to better judge the performance of the models, Matthew's correlation coefficient (MCC) was used in the performance analysis. A connection between -1 (difference between observation and prediction) and +1

(perfect prediction) is provided by MCC, commonly known as the phi coefficient. A coefficient of 0 means that it is an average random prediction.

3.6.6. Mean squared error (MSE)

For observing and evaluating the performance of time-based models, Mean Squared Error (MSE) is usually the popular metric. MSE emphasizes Bigger errors are highlighted more in MSE. This signifies the use of MSE in scenarios like FECG where even little alterations in the signals could ultimately equate to considerable impact of the overall FECG signal. When the error differences are squared off then this leads to MSE to penalize bigger errors as compared to little errors. Also, MSE (being a continuous function) is suitable in terms of mathematics for model training optimization. Overall, it is a reliable metric which observes the estimation error. Eq. 3.11 displays the formula of MSE.

$$MSE = \frac{1}{n} \sum_{i=1}^n (y_i - y_i^o)^2 \quad (3.11)$$

Here: n = number of data points. y_i = observed values. y_i^o = predicted values.

Root Mean Squared Error (RMSE) is another statistic applied in this study. RMSE is usually selected over MSE when it is to be found if a model fits a dataset. Eq. 3.12 displays the formula of RMSE. Average difference between anticipated data and actual data is “x” if RMSE is “x”.

$$RMSE = \sqrt{MSE} \quad (3.12)$$

3.7. Dataset Overview

The Omniview-SisPorto 2.0 software [23] produced the dataset utilized in this research study (for Phase A and Phase B), which was obtained from [19]. Since the SisPorto 2.0 program carefully adhered to the International Federation of Gynecology and Obstetrics (FIGO) criteria while performing an automated analysis of cardiotocogram (for both ante- and intrapartum tracings). The final dataset also met the FIGO requirements. The dataset includes 21 characteristics (as shown in Table 3.2), which are divided into three groups

(Normal, Suspect, and Pathologic), 10 diagnostic classes (as shown in Table 3.3), and 2126 CTG recordings. The code for fetal status in the CTG dataset is NSP and the code for morphological pattern classes is CLASS. Three field relevant related specialists had independently validated all features and classes [23].

Table 3.2. The Features or Predictors in the CTG Dataset

Features (or Predictors)	Symbol in Dataset	Explanation
Fetal Heart Rate (FHR) Baseline	LB	FHR measured in the number of heart beats per min
Accelerations	AC	Number of AC per second
Fetal Movements	FM	Number of FM per second
Uterine Contractions	UC	Number of UC per second
Light Decelerations	DL	Number of DL per second
Severe Decelerations	DS	Number of DS per second
Prolonged Decelerations	DP	Number of DP per second
Abnormal Short-Term Variability	ASTV	Percentage of time with ASTV
Mean Short-Term Variability	MSTV	Mean value of STV
Abnormal Long-Term Variability	ALTV	Percentage of time with ALTV
Mean Long-Term Variability	MLTV	Mean value of LTV
Width of FHR Histogram	Width	-
Minimum of FHR Histogram	MIN	-
Maximum of FHR Histogram	MAX	-
Number of Histogram Peaks	NMAX	-
Number of Histogram Zeros	NZEROS	-
Histogram Mode	Mode	-
Histogram Mean	Mean	-
Histogram Median	Median	-
Histogram Variance	Variance	-
Histogram Tendency	Tendency	Left asymmetric, Symmetric and Right asymmetric

Table 3.3. The CTG Related Classes of Morphological Patterns in the CTG Dataset

Classes of Morphological Pattern	Symbol in Dataset	Explanation
Calm Sleep	A	-
REM Sleep	B	Rapid Eye Movement
Calm Vigilance	C	-
Active Vigilance	D	-
Shift Pattern	E	Shifts between Class A or Class SUSP
Accelerative/Decelerative Pattern	AD	Fetus Stress Situation
Decelerative Pattern	DE	Vagal Stimulation
Largely Decelerative Pattern	LD	-
Flat-Sinusoidal Pattern	FS	Pathological State
Suspect Pattern	SUSP	-

According to FIGO guidelines [1], fetal condition is assessed and classified by the following criteria: The FHR evaluation is Normal if the baseline heart rate is between 110 and 160 beats per minute, variability is between 5 and 25 beats per minute, and there are no decelerations. This indicates that there is no need for intervention because the fetus has neither hypoxia nor acidosis. The FHR assessment is pathological if the baseline heart rate is less than 100 beats per minute, the variability has been reduced for more than 50 minutes (or increased for more than 30 minutes or there is a sinusoidal pattern for more than 30 minutes), or if decelerations are late or prolonged for more than 30 mins. In this situation, there is a great likelihood that the fetus may experience hypoxia and acidosis, thus prompt treatment is necessary. If there is an absence of at least one characteristic associated with a normal case, while no pathological characteristic is observed, then the FHR evaluation criterion is suspicious (or suspect). In this situation, there is a low chance that the fetus would have hypoxia or acidosis, thus ongoing observation is necessary. SMOTE was used to address the imbalanced CTG dataset, which was evident from Figure 1 and Figure 2. As it can be observed that Normal case for fetal status case and Class B for CTG morphological pattern case are majority classes that greatly skews the CTG dataset. By interpolating minority class entries into the feature space of the data, SMOTE creates synthetic entries in the dataset. Between a minority sample and its k-neighbors, new entries are created due to SMOTE. However, it should be noted that SMOTE was used in Phase A but not in Phase B. The reason for this selection was to see the performance of two different proposed models on the same CTG dataset.

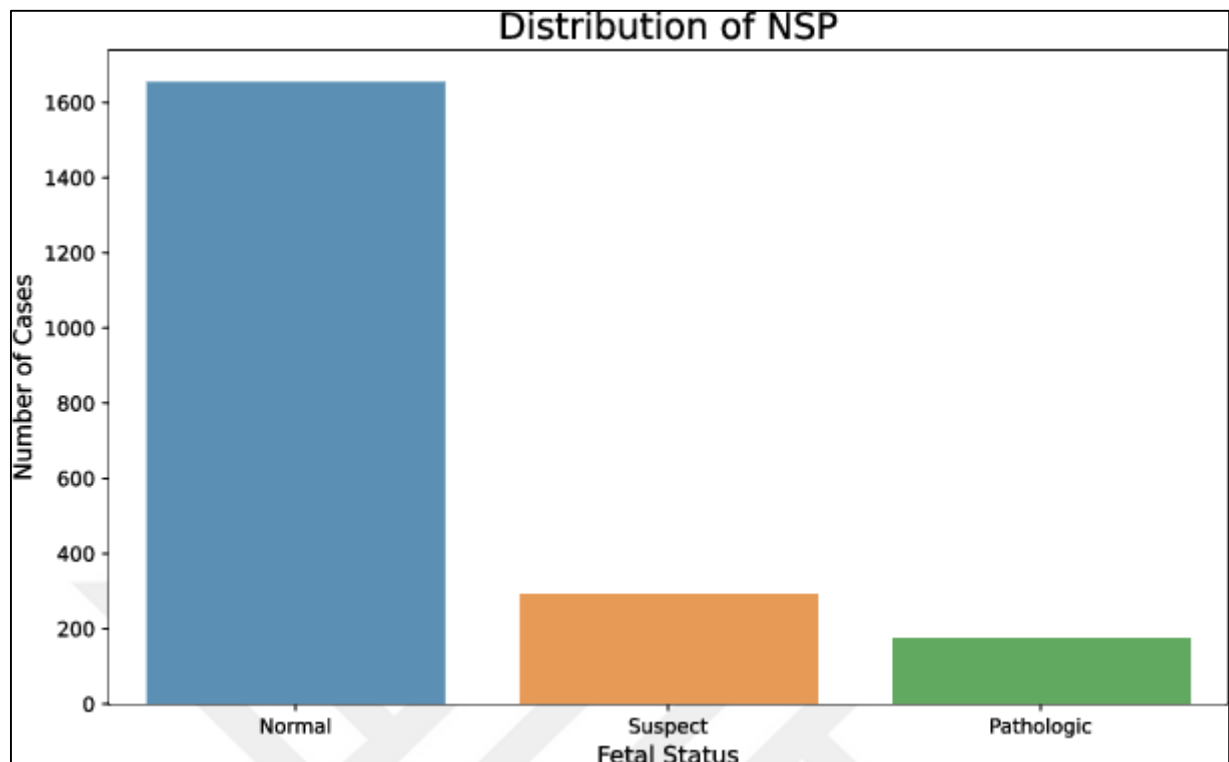


Figure 3.1. Imbalance Characteristics of the CTG Dataset in the Fetal Status Case

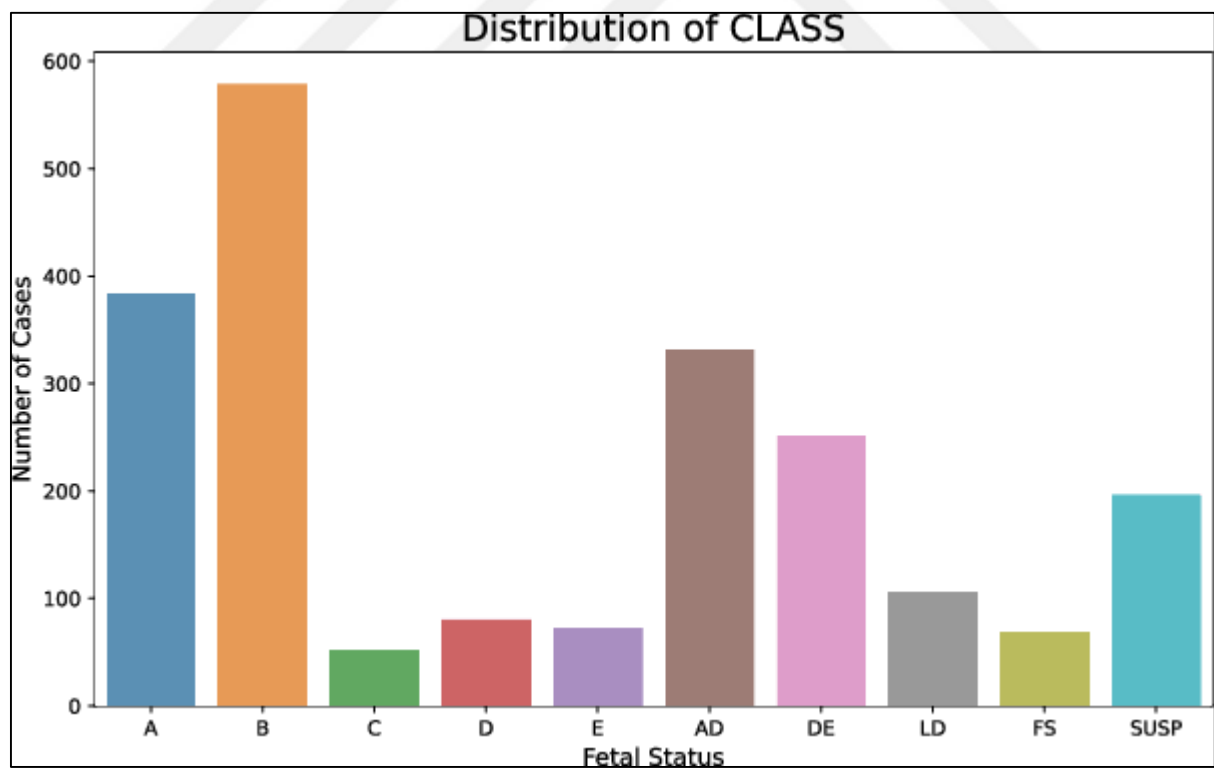


Figure 3.2. Imbalance Characteristics of the CTG Dataset in the Morphological Pattern Classes

3.7.1. Correlation analysis

In the domains of data analysis and machine learning, correlation is a fundamental concept that reveals unobserved connections between variables. Understanding and utilizing correlation may considerably improve the accuracy and effectiveness of models when it comes to categorization jobs. In this article, we explore the idea of correlation in categorization data, its significance, measurement techniques, and effects on prediction models. The statistical link between two or more variables is measured by correlation. These factors may be characteristics or traits that affect how a class or category performs in the context of categorization. Finding correlations allows us to acquire understanding of how different variables interact, which may help us spot patterns and trends that may guide our decision-making. Building models that can reliably predict the class or category of a given input is the ultimate objective of classification problems. Correlation is crucial to reaching this objective for a number of reasons, that are given as follows:

Correlation analysis assists in choosing the features that are most pertinent for categorization. Overfitting may result from highly linked characteristics carrying duplicate information. We may increase the effectiveness and generalization of the model by choosing the appropriate characteristics.

Discovering correlations can lead to the development of fresh features or transformations that improve class separability. This could result in better model performance.

Correlation insights can offer a greater comprehension of how specific features affect the categorization result. Building confidence in machine learning models requires this openness.

Variables that correlate with one another are dependent on one another. A classification model is not directly impacted by correlation; however, the existence of a large number of correlated variables in a dataset might have an indirect impact on the model by adding redundant variables (features) that can lower the classification model's quality.

Eq. 3.13 gives the Spearman correlation formula if r is the Spearman's rank correlation coefficient, d_i^2 is the square of the difference in the ranks of two variables for each pair, and n is the number of pairings.

$$\rho = 1 = \frac{6 \sum d_i^2}{n(n^2 - 1)} \quad (3.13)$$

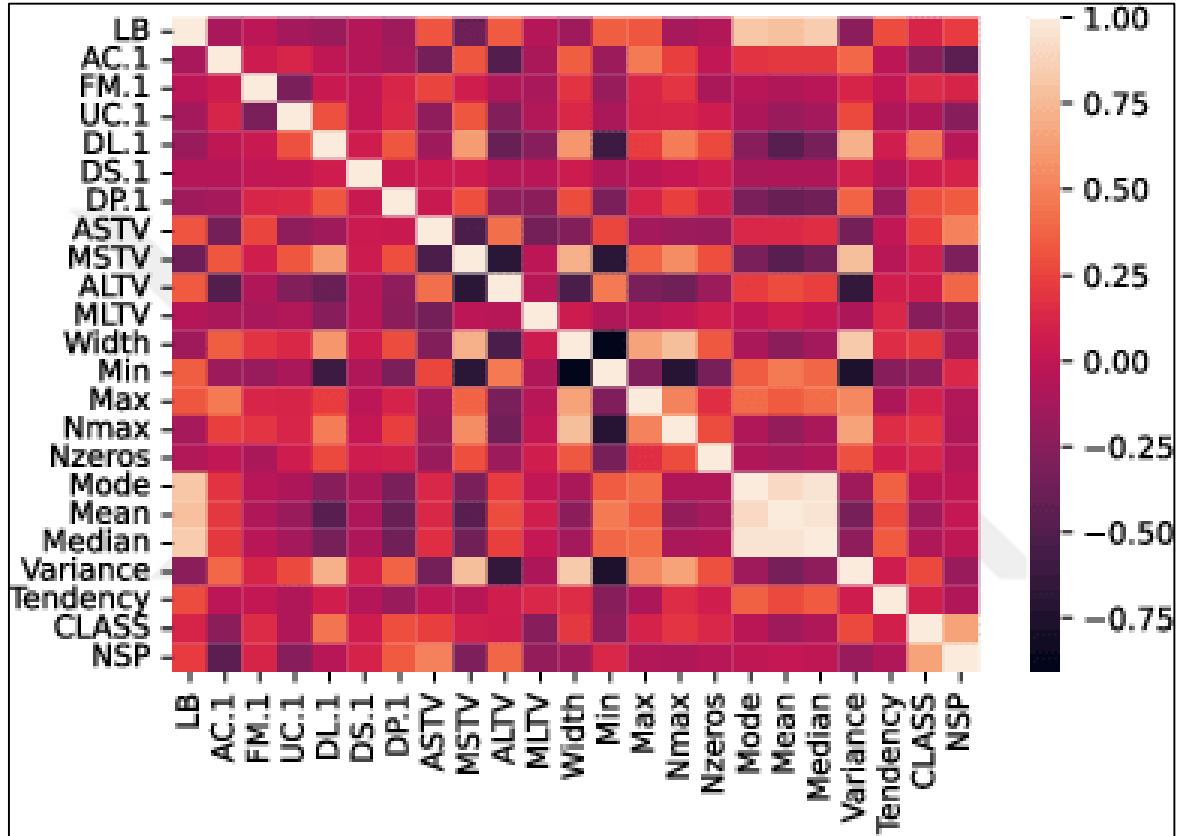


Figure 3.3. Correlation Heatmap of the CTG Dataset using Spearman Correlation

A substantial (>0.7) connection between the FHR Baseline (LB) and the FHR histogram descriptors (Mode, Mean, and Median), as well as between the width, minimum, maximum, and number of histogram peaks, was seen in the Spearman correlation heatmap (Figure 3.3). The Mode, Mean, and Median FHR histogram descriptors have some internal correlations. This was further supported by [37], where the author employed correlation analysis (CA) to establish a connection between the FHR histograms and explanatory data analysis (EDA) to establish a connection between the accelerations and decelerations of uterine contractions. The ability of the model to forecast accurately may suffer as a result of the similarities. This offered yet another justification for using a feature extraction method to eliminate extraneous

features. The simulation phase will address the problem caused by strongly (positive and negative) linked duplicate features.

Almost similar correlation result (as shown in figure 3.4) is shown if Kendall's correlation is implemented on the CTG dataset. The reason to use another correlation method to assess the given CTG dataset was to make sure that the proposed models (discussed later in section 3.8) do suffer from the case where similar features of a dataset negatively affect the overall performance of the model. The Kendall correlation equation is given in Eq. 3.14, where τ is the Kendall's correlation coefficient, n_0 is the total number of pairs, n_1 is the number of pairs tied in the first variable, n_2 is the number of pairs tied in the second variable, n_c is the number of concordant pairs, and n_d is the number of discordant pairs.

$$\tau = \frac{n_c - n_d}{\sqrt{(n_0 - n_1)(n_0 - n_2)}} \quad (3.14)$$

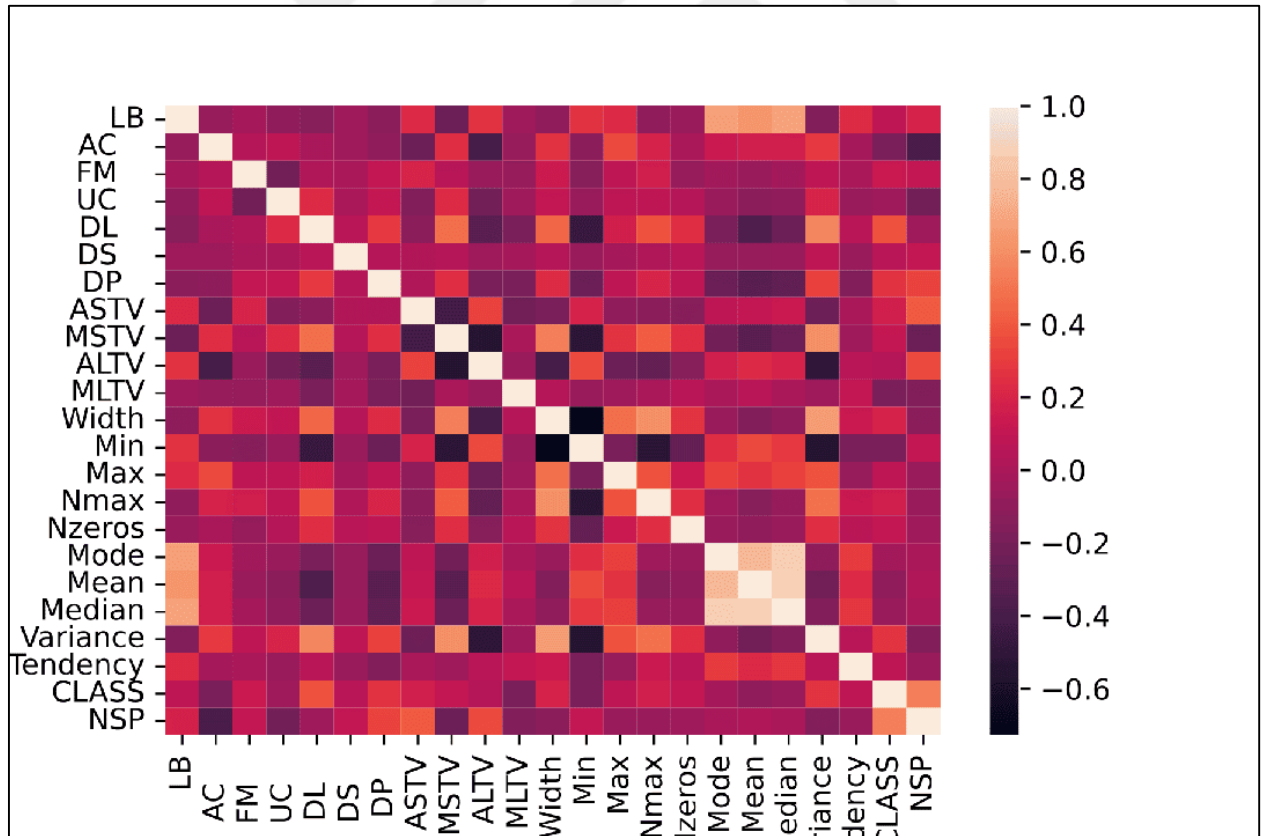


Figure 3.4. Correlation Heatmap of the CTG Dataset using Kendall Correlation

3.7.2. Data preprocessing

In the realm of data analysis and machine learning, there is no clear path from raw data to insightful conclusions or precise forecasts. Instead, it entails preprocessing, an important first step. Preprocessing, which is frequently seen as the cornerstone of successful data-driven initiatives, includes a number of methods for converting unstructured, dirty data into a clean, organized format that can be used by analytical and machine learning models. This vital procedure has a substantial bearing on the accuracy and dependability of the outcomes, making it an essential component of any data-centric endeavor. A collection of methods known as preprocessing are used to get data ready for further analysis or modeling. Raw data is frequently full of errors, missing numbers, outliers, noise, and inconsistencies since it is gathered from a variety of sources. Preprocessing takes care of these issues to make sure the data is ready for additional analysis. Important preprocessing steps are as follow:

Data cleaning includes locating and fixing mistakes in the dataset. It involves dealing with missing values using techniques like imputation, getting rid of duplicate entries, and fixing discrepancies.

Techniques for transforming data are used to change the scale or distribution of variables. Common transformations include normalization, which decreases the impact of outliers by scaling data to a common range, and log transformation.

Finding the best features from a dataset is important. Not all characteristics are equally informative. In order to reduce dimensionality and simplify the model, feature selection entails finding and maintaining just the most pertinent characteristics. The most important variance in the data is captured by new features that are produced using feature extraction techniques like Principal Component Analysis (PCA).

Since a lot of machine learning algorithms require numerical input, categorical data, such text labels, must be transformed. Techniques like one-hot encoding, which transforms category data into binary vectors, may be used in this.

Outliers have a big impact on model performance. Truncation is one of the popular preprocessing techniques that aid in controlling the influence of outliers on the study.

Integration guarantees consistency and compatibility while dealing with multiple datasets. Datasets are frequently combined, joined, or concatenated in integration procedures.

The significance of preprocessing is that the foundation for precise analysis and modeling is laid by efficient preprocessing. Due to skewed or noisy input, machine learning models may perform poorly without sufficient preprocessing. Raw data might be used to derive inaccurate or deceptive conclusions, which would result in poor decision-making. Preprocessing also conserves critical computing resources. The data may be managed better, and machine learning model training becomes more effective by removing useless characteristics and lowering noise. As for the challenges regarding preprocessing are concerned, although preprocessing is necessary, however there are difficulties involved. Common challenges include choosing the best methods for a certain dataset, preventing overfitting during preprocessing, and balancing feature selection with the preservation of important information. The unsung hero of data analysis and machine learning is preprocessing. It provides the framework on which precise insights and predictive models are constructed. For workers in industries ranging from banking and healthcare to marketing and technology, learning preprocessing techniques is becoming increasingly important as the data landscape continues to change. The raw potential of data is polished into a useful asset that supports informed decision-making and innovation through thorough and intentional preparation.

The CTG dataset [19] underwent preprocessing by having several empty rows and a few information-free columns removed. The dataset predictor or feature columns also do not include any Not a Number (NaN) entries. In addition to the characteristics or features given in Table 3.2, the CTG dataset [19] also had the feature DR (Repetitive decelerations); however, it was eliminated due to the fact that the authors implemented chi-squared test on all of the dataset features and found that $p(K-W) = 1$ was discovered for DR feature. This elimination was supported by [40] as well. The desired output for fetal state was NSP (with 3 classes), while the target output for the CTG morphological pattern was CLASS (with 10 classes). The “z-score” was used to standardize the dataset. The formula (Eq. 3.15) for the z-score has μ as the sample mean and σ as the sample standard deviation.

$$z = \frac{x - \mu}{\sigma} \quad (3.15)$$

Missing information in CTG recordings can significantly affect how fetal well-being is interpreted and can result in less-than-ideal labor and delivery management decisions. The following problems can arise as a result of missing data:

An insufficient evaluation of fetal health since the CTG recording provides essential data on fetal heart rate and uterine contractions. As a result, opportunities to identify early fetal distress or hypoxia may be missed.

A misreading of the CTG pattern. As a result, unnecessary procedures like emergency cesarean sections could be performed when they are not required.

A delay in properly managing labor (and delivery) and making decisions. As a result, this can be detrimental to both the mother and the fetal health.

At the preprocessing part, missing values related problem can be addressed. Preprocessing the CTG dataset is therefore very important. Research in [18] describes a technique for filling in missing data that uses two iterative phases. An adaptive dictionary is employed in the initial “reconstruction step” to rebuild the signal or data that results in the estimate of the missing data. On the basis of the reconstructed signal (in the first step), a new dictionary is then created in the second step using the KSVD (k-singular value decomposition) technique. Repeating these two actions will bring about convergence. The program produced positive results, especially for repeated missing samples. The automatic analysis in the SisPorto 2.0 program produced the CTG dataset [19] that was taken into consideration for this study (for Phase A and Phase B). The issue with missing data was resolved by the above-mentioned program by default and hence the proposed models do not suffer from problem of the missing data in a dataset.

3.7.3. Dataset for Phase C

Andreotti et al [20] proposed and deployed a synthetic ECG database called The FECGSYN (v1.0.0). It includes dipole simulations of the mother’s and fetal hearts in various spatial and magnitude configurations. Each abdominal signal component is considered as a separate

generator, whose input signal is transferred onto the detector sites, fetal-maternal mixes are produced. Figure 3.5 displays the electrode configuration and the location of the source for creating artificial signals. As a result of the realistic placement of the electrodes, each signal source can have a unique waveform file in this database. Ten distinct simulated pregnancies are present in the modeled volume conductor, which is sampled from 34 channels (32 abdominal and 2 maternal ECG reference channels). Physionet was used as a source for the FECGSYN database that was in the MIT waveform format where signal, header and synthetic QRS data is stored in WFDB (WaveForm DataBase).

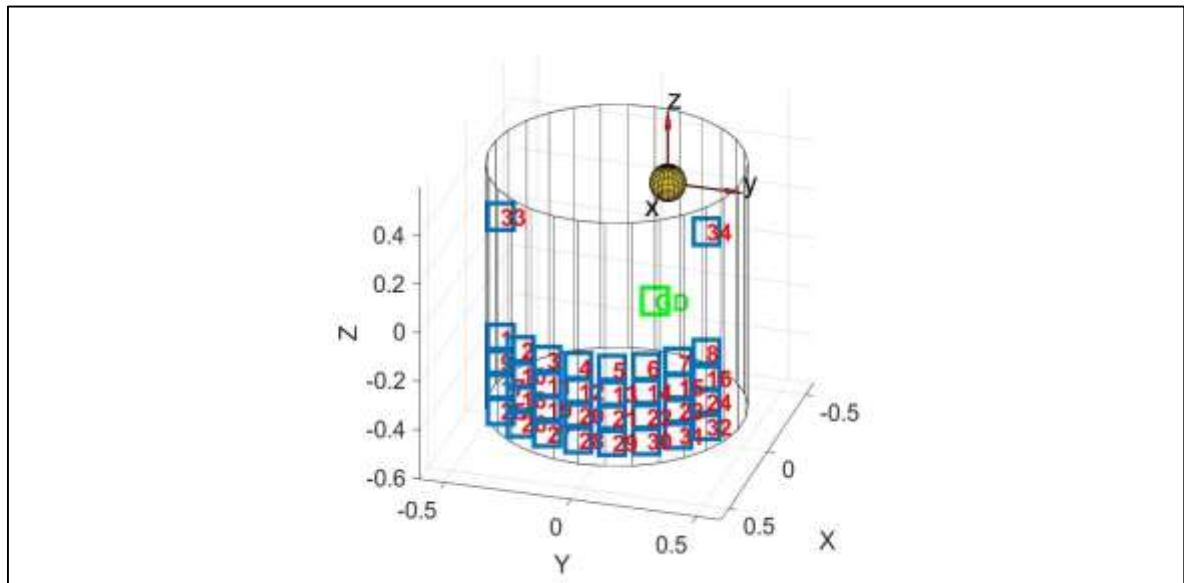


Figure 3.5. FECGSYN Generation [20]

There are many scenarios within the FECGSYN database. For each synthetic subject, 7 distinct physiological events are proposed as displayed in Table 3.4.

Table 3.4. FECGSYN Dataset Overview

Case Code	Description
Baseline Case	Abdominal mixed signal with neither noise nor highlights
Case 0	Baseline signal with noise but no events
Case 1	Fetal movement event with noise
Case 2	Accelerations or decelerations for MHR and FHR with noise
Case 3	UC with noise
Case 4	For both fetus and mother, Ectopic beats with noise
Case 5	Extra FECG with noise. This also means there is a twin pregnancy

Moreover, for each set of parameters, there are 5 distinct ranks of noise (SNR levels ranges are 00, 03, 06, 09, 12 dB), with five repeats (l1 to l5). All 1750 synthetic signals are generated from 10 simulated pregnancies, 7 cases, 5 SNR levels, and 5 repeats, yielding 145.8 hours of data in multichannel form and 1.1 million peaks of fetal activity. Every iteration lasted for 5 minutes and used a 250 Hz sampling rate.

3.8. Methodology

As discussed in the introduction section, there are three phases in this research. All these phases have different methodologies. Phase A and Phase B use the CTG dataset [19] while Phase C use the FECGSYN dataset [20].

3.8.1. Methodology for Phase A

The proposed model in Phase A was implemented in such a manner that it follows the STROBE guidelines [56]. The input matrix of the CTG dataset was made larger by applying SMOTE, increasing its size to 4965. A model that was influenced by feature extraction (with the help of autoencoders) and feature selection (with the help of RFE) was presented for this dataset, which is relatively bigger. Both FS as well as FE have ability to change the input, but FS preserves input and eliminates predictors (or features) according to their ranks, while FE adds new predictors (or features) at the same dynamically eliminating the unwanted ones. Duplicate items and dataset dimensionality were to be decreased by the suggested model. Figure 3.6 depicts the proposed algorithm.

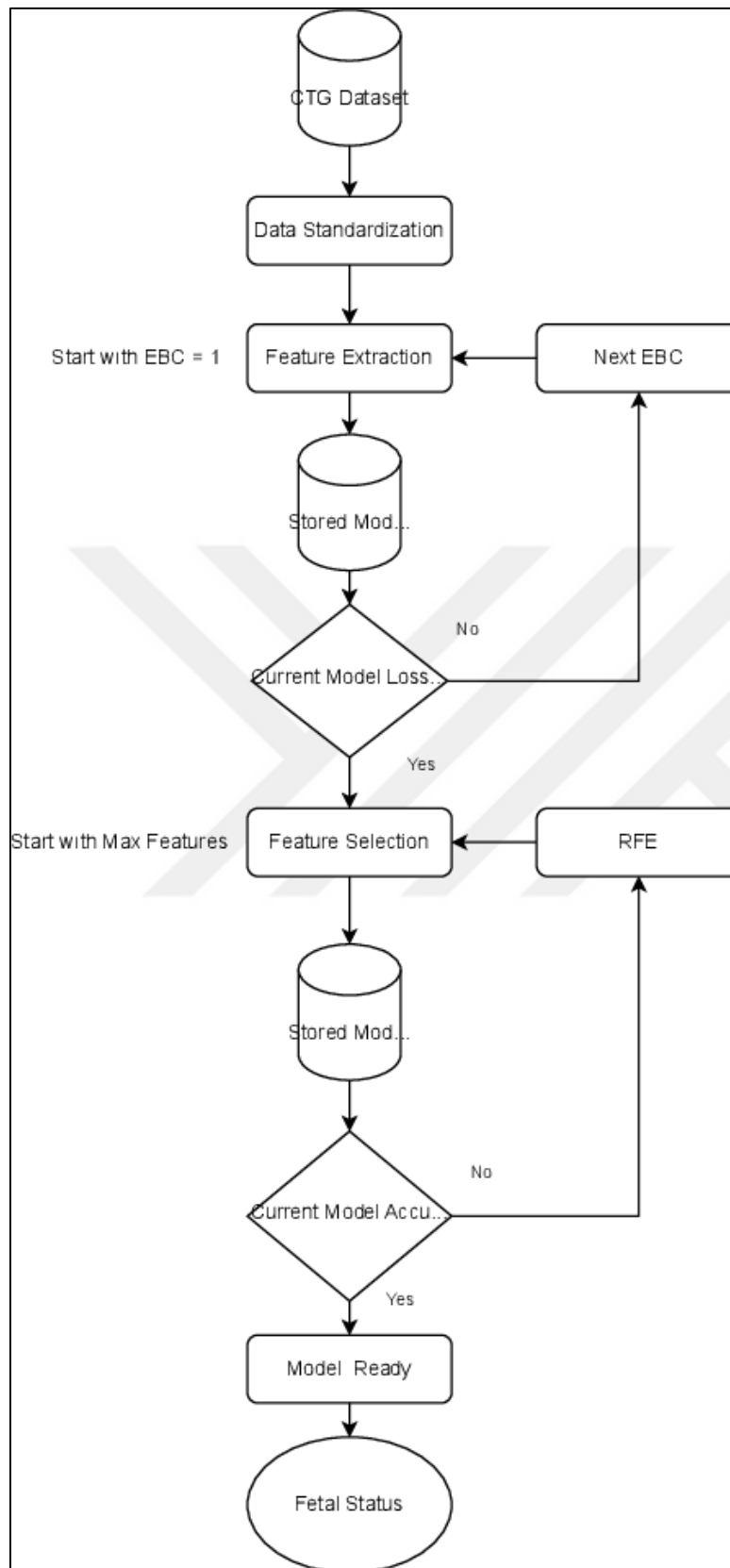


Figure 3.6. Proposed Algorithm for Phase A

To resolve the encoder bottleneck information (EBI) was the main objective of the FE module. Eq. 3.16 provides number of encoder bottleneck neurons (EBNs) in the bottleneck layer.

$$EBN = \frac{OFmax}{EB_c} \quad (3.16)$$

Here OFmax is max number of original predictors (or features), while EB_c is encoder bottleneck coefficient. Depending on how the algorithm performs, the value of EB_c ranges from 1 to 2. EBC initial value was set to 1. The results demonstrated that low-quality data reconstructions were produced when $EB_c > 2$. However, $EB_c = 1.5$ was shown to be a suitable balance between reduction in the dimension and maintaining lowest amount of predictors (or features) need that could lead to an improved classification. Traditionally, choosing the EBI dimensions is not a fixed approach. Reducing dataset dimensions was FE module's secondary job. New characteristics that represented the old CTG dataset were present in the new dataset. RF was utilized as an estimation element for RFE in FS module. Since the RFE relies on feature importance rather than feature type, it has the benefit of being indifferent about the type of input features. The fetal state and CTG morphological pattern were then classified using the final model with RF as a base classifier. The final model, at the end of the FS module, was also passed to a BO module when it reached the “Model Ready” stage. BO treats each issue as a black box and iteratively seeks best value with fewest possible iterations. As BO aims to strike a balance between exploitation (coming to the optimal decision based on knowledge) and exploration (gathering additional information) [51].

The split criterion, class weights, max predictor (or feature) number for a split, and estimators which are the tree amount in a forest, were among the hyperparameters that were adjusted for RF in order to achieve the best result. SisPorto 2.0 [23] was used to annotate the CTG dataset; as a result, feature extraction was used to create the predictors and features. Although, Figure 3.6 shows that second FE was carried out after SMOTE algorithm was put into use (which was done to address the class imbalance in the CTG dataset). The fact that the CTG dataset, after SMOTE, was now relatively bigger and thus there was a chance that a part of newer dataset was redundant. Moreover, larger CTG dataset would raise computing

expenses leading to the necessity for a second feature extraction in the proposed model for Phase A.

3.8.2. Methodology for Phase B

The proposed model in Phase B was implemented only on the NSP target output of the CTG dataset [19]. This was done to make a focused algorithm that dealt with only the fetal condition. The proposed algorithm for Phase B is as follows: Let the original number of features be F , and the original feature set be $OF = F_i$, where $i = 1, 2, \dots, F$. Let Clf be a classifier to calculate and list the rank of the features.

1. Iteration starts.
2. Let $Clf = [R_i]$, where $i = 1, 2, 3, \dots, F$. R is the rank of all the features of the dataset. Initialize and populate a model accuracy set (Clf Acc) that will contain the model accuracies, appended per iteration. Initialize and populate a model MCC set (Clf MCC) that will contain the model accuracies, appended per iteration. Let MCC_{max} be the highest MCC value in a Clf MCC set.
3. Perform RFE (elimination of least important feature) over Clf rank set until conditions in the step 4 and step 5 are met:
 - a. If conditions are met, then stop elimination of features and save the last Clf rank set [Step 4 will produce the last least important feature to be eliminated and Step 5 will produce the most important feature]. Goto Step 6.
 - b. Else continue with Step 3.
4. For every two consecutive entries in the Clf Acc set:
 - a. If two consecutive sets have positive (and/or zero) percentage increase in accuracies, then continue with step 4.
 - b. If two consecutive sets have a negative percentage increase in accuracies, then stop the elimination. The least important feature in the Clf rank set is highlighted. Goto step 5.
5. Calculate the MCC for each Clf rank set for i th iteration:
 - a. If $MCC_i \geq MCC_{max}$ then Goto Step 3 (the most important feature in the Clf rank set is highlighted).
 - b. Else continue with step 3.
6. List out all hyperparameters and number of features that produced the optimum result.
7. Iteration stops.

3.8.3. Methodology for Phase C

For the case involving the FECGSYN dataset, its related suggested methodology is displayed in Figure 3.7. The artificial signals generated from the FECGSYN software were first preprocessed and converted into machine learning centric type. Also, a sliding window (having a window size of fifty, the inspiration of this selection was acquired from [49, 50]) was implemented which worked with fifty samples for input that was in turn needed to estimate the 51st point (this point was considered as the output for that event). Later the sliding window was shifted to find the next point. These iterations only stopped when all points had gone through the window. Hundred epochs were considered as the stopping condition for LSTM training stage. The trained model was tested by testing part of the dataset to display performance results. Anomaly was highlighted if MSE value was larger than a fixed Threshold. By this mechanism the anomalies within those ECG based signals were observed and highlighted. The effects of the threshold selection on the anomaly detection mechanism were also observed by changing the threshold value for different scenarios.

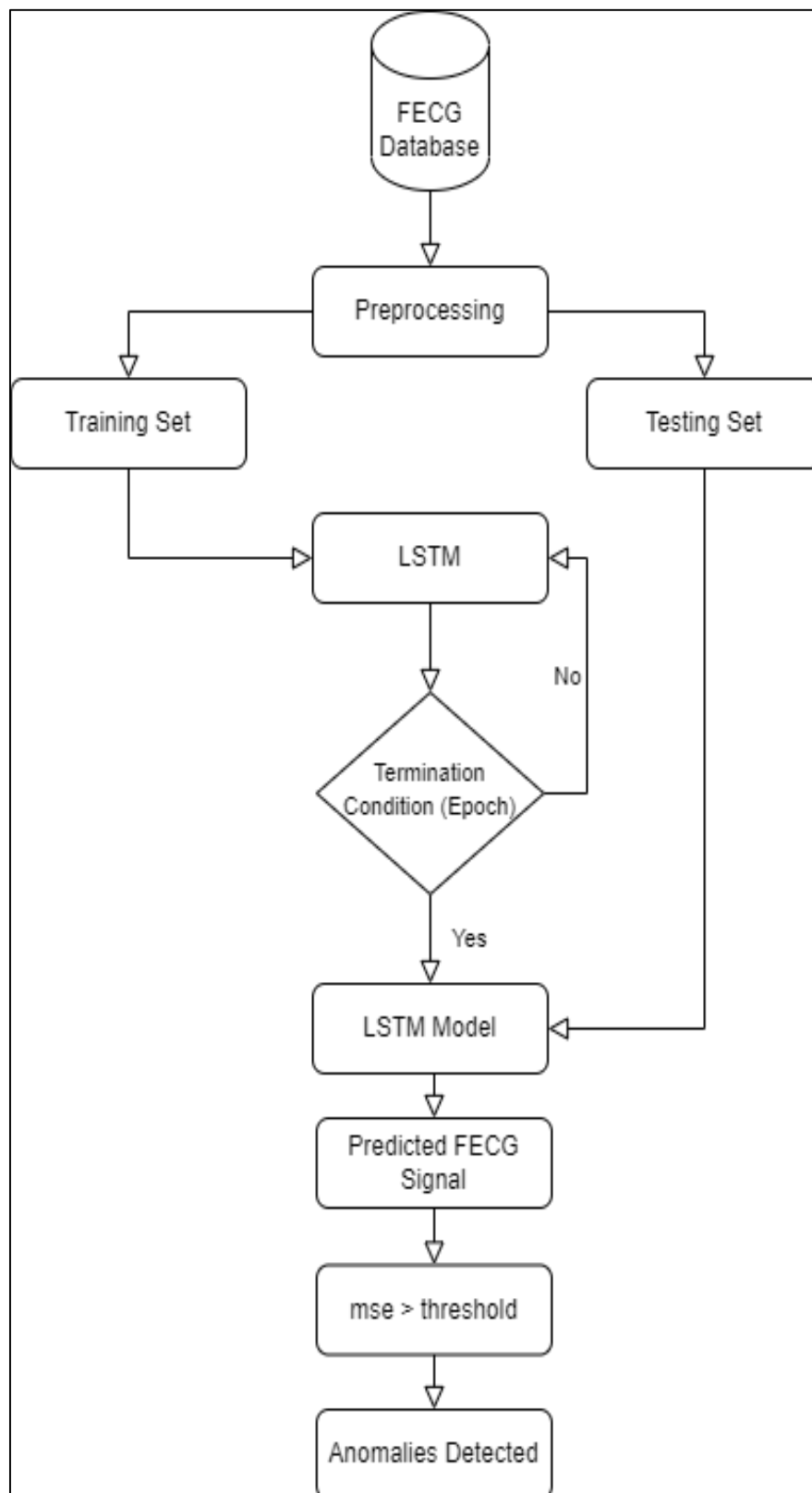


Figure 3.7. Flowchart of the Proposed Algorithm for Phase C



4. RESULTS

As there were three different phases proposed in this research study, there are three types of results in this thesis study as well. The subsequent subsections contain the results obtained for all the three phases (Phase A, Phase B, and Phase C). For all phases, Python environment was used. The reason to use Python was due to its prominence in the field of machine learning as it has an extensive library ecosystem, which includes TensorFlow, PyTorch, scikit-learn, and keras. These libraries facilitate an effective model development. Moreover, Python has an intuitive syntax that enables an easy expression of complex algorithms. Most importantly Python has a vibrant community that offers a wealth of learning resources. Moreover, Python has powerful data manipulation abilities with libraries like pandas; that provide seamless integration with various technologies.

4.1. Results of Phase A

The simulations for Phase A were carried out in the Python 3.8 environment. Two parts made up the simulations. The part one includes fetal status component from CTG dataset [19], whereas part two includes morphological pattern component. Moreover, for the purpose of measuring the performance comparison of proposed model (in phase A), simple normal RF (excluding the implementation of the proposed model of phase A) was also implemented and later compared with the proposed model. For both parts, 75:25 was established as the Training to testing ratio. Moreover, another reason to use RF as a base classifier is that it can naturally handle multiclass classification.

4.1.1. Fetal status classification in Phase A

Proposed model for the fetal status classification was evaluated utilizing performance indicators listed in Table 3.1. The resulting performance analysis is shown in Table 4.1. Moreover, Table 4.2 also includes performance analysis from the scenario where only RF, excluding the implementation of the proposed model (PM), was employed for comparison.

Table 4.1. Performance Evaluation of the Proposed Fetal Status Model

Fetal Status	Precision	Recall	Specificity	F1-Score
Normal	0.9844	0.9313	0.9928	0.9571
Suspect	0.9247	0.9800	0.9619	0.9515
Pathological	0.9907	0.9861	0.9950	0.9884

Table 4.2. Performance Evaluation of RF (without PM) for Fetal Status

Fetal Status	Precision	Recall	Specificity	F1-Score
Normal	0.9527	0.9829	0.8360	0.9675
Suspect	0.8333	0.6944	0.9782	0.7575
Pathological	0.9183	0.9000	0.9917	0.9090

The best result of suggested model for CTG fetal status for phase A had thirteen predictors (or features), and its model accuracy was 96.62%. On the other hand, an accuracy of 93.61% was attained when only Random Forest was applied on same CTG dataset (which included the complete twenty-one predictors or features). Table 4.3 displays confusion matrix for suggested model of phase A for fetal status. The items of the confusion matrix are shown as percentages to make comparisons easier. Moreover, Table 4.4 displays a confusion matrix from the scenario where only simple Random Forest (excluding suggest phase A model) was utilized.

Table 4.3. Confusion Matrix of The Proposed Fetal Status Model

		Predicted		
		Normal	Suspect	Pathological
Actual	Normal	93.1%	6.4%	0.5%
	Suspect	1.5%	98.0%	0.5%
	Pathological	0.0%	1.4%	98.6%

Table 4.4. Confusion Matrix of Basic RF for Fetal Status

		Predicted		
		Normal	Suspect	Pathological
Actual	Normal	98.3%	1.5%	0.2%
	Suspect	26.4%	69.4%	4.2%
	Pathological	2.0%	8.0%	90.0%

According to Figure 3.6, the “Model Ready” stage already has Bayesian Optimization (BO) implemented in it, impact of BO in this proposed algorithm is discussed at a later stage. Moreover, the results shown in Table 4.1 and Table 4.3 already had the involvement of BO.

For each of the 3 classes (Class 1 has the Normal values, Class 2 has the Suspect values, and Class 3 has the Pathologic values), ROC (along with the AUC) curve and PR curve were assessed separately, as shown in Figure 4.1 and Figure 4.2, respectively.

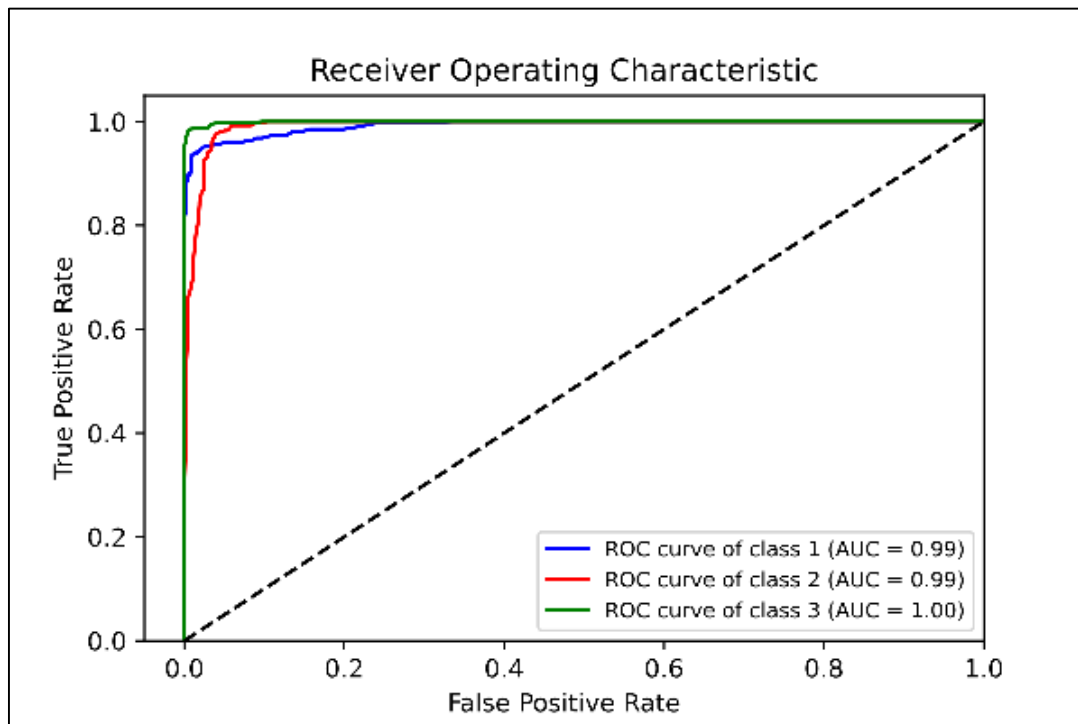


Figure 4.1. ROC Curve (AUC) of the Proposed Fetal Status Model

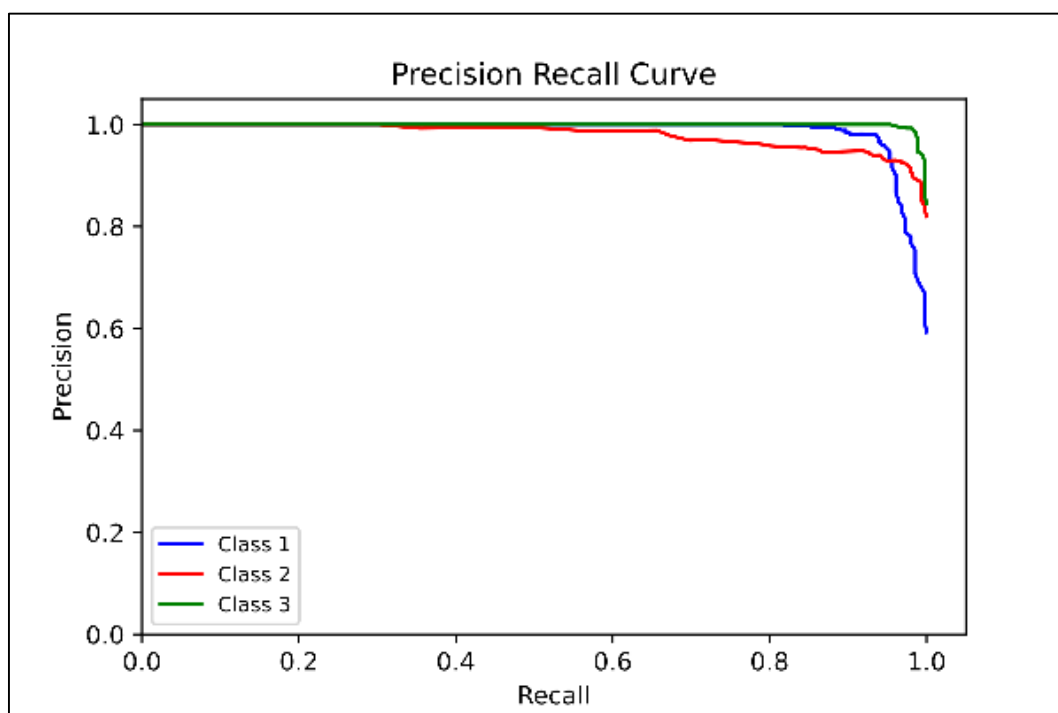


Figure 4.2. PR Curve of the Proposed Fetal Status Model

Figure 4.3 shows the variance in model accuracy over the course of the complete iterations of the suggested phase A model for fetal status. When 13 features were selected then the maximum accuracy (96.62%) was achieved.

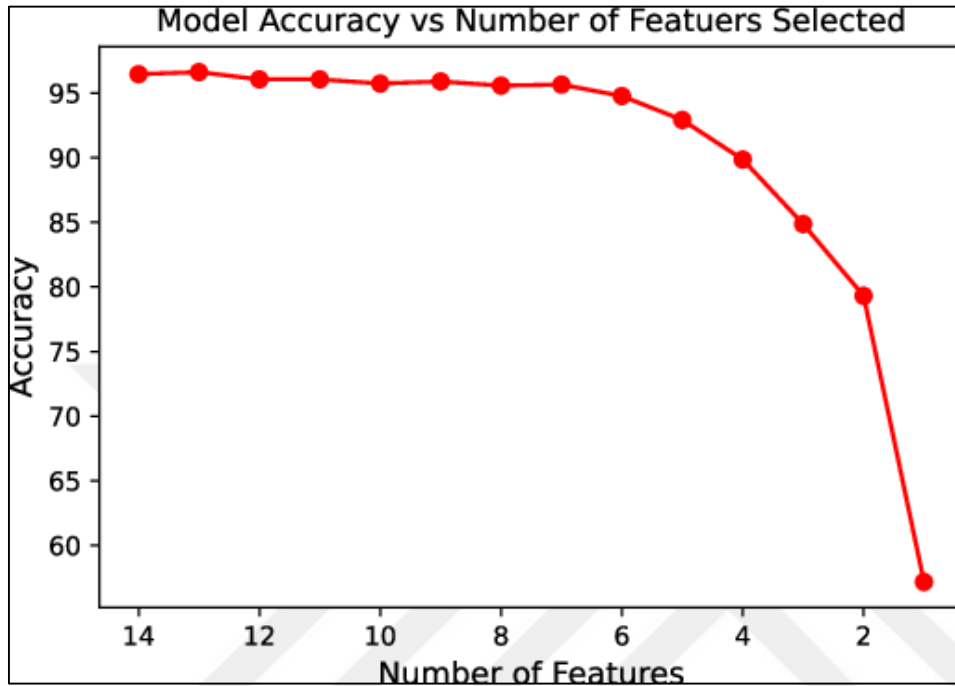


Figure 4.3. Relationship between Model Accuracy and the Total Number of Features Selected for Fetal Status in Phase A

4.1.2. CTG Morphological Pattern Classification in Phase A

The suggested phase A model for the CTG morphological pattern classification was evaluated utilizing performance indicators listed in Table 3.1. The resulting performance analysis is displayed in Table 4.5. Moreover, Table 4.6 also includes performance analysis from the scenario where only RF, excluding the implementation of the proposed model (PM), was employed for comparison.

Table 4.5. Performance Evaluation of the Proposed CTG Morphological Pattern

Morphology Pattern	Precision	Recall	Specificity	F1-Score
A	0.9389	0.8092	0.9938	0.8692
B	0.9565	0.8859	0.9953	0.9198
C	0.9459	0.9929	0.9938	0.9688
D	0.9934	0.9934	0.9992	0.9934
E	0.9281	0.9872	0.9907	0.9567
AD	0.9302	0.9160	0.9931	0.9230
DE	0.9225	1.0000	0.9916	0.9597
LD	1.0000	0.9870	1.0000	0.9934
FS	0.9520	0.9720	0.9946	0.9619
SUSP	0.9225	0.9562	0.9916	0.9390

Table 4.6. Performance Evaluation of RF (without PM) for CTG Morphology

Morphology Pattern	Precision	Recall	Specificity	F1-Score
A	0.8073	0.9166	0.9518	0.8585
B	0.8758	0.9241	0.9505	0.8993
C	1.0000	0.6363	1.0000	0.7777
D	0.8750	0.7000	0.9960	0.7777
E	1.0000	0.4583	1.0000	0.6285
AD	0.8658	0.8875	0.9756	0.8765
DE	0.9473	0.8307	0.9935	0.8852
LD	1.0000	1.0000	1.0000	1.0000
FS	0.8823	0.7894	0.9961	0.8333
SUSP	0.7959	0.9512	0.9796	0.8666

The suggested phase A model 94.96% was the accuracy for CTG morphological pattern, when fourteen features were selected by the model. While an accuracy of 87.22% was attained using basic RF (excluding the proposed model) on the same CTG dataset (with all 21 features included). Table 4.7 displays confusion matrix of suggested phase A model for CTG morphological pattern. The items in confusion matrix are shown as percentages to make comparisons easier. Moreover, Table 4.8 displays a confusion matrix from the scenario where only simple Random Forest (excluding the suggested phase A model) was utilized. Additionally, the blank entries in Table 4.7 and Table 4.8 represent an entry of 0.0.

Table 4.7. Confusion Matrix of The Proposed CTG Morphological Pattern Model

		Predicted									
		A	B	C	D	E	AD	DE	LD	FS	SUSP
Actual	A	80.9	1.3	4.6		5.9	0.7	1.3		2	3.3
	B	4	88.6	0.7		2	4.7				
	C		0.7	99.3							
	D				99.3		0.7				
	E					98.7					1.3
	AD		2.3		0.8		91.6	5.3			
	DE							100			
	LD							1.3	98.7		
	FS									97.2	2.8
	SUSP	1.5								2.9	95.6

Table 4.8. Confusion Matrix of basic RF for CTG Morphological Pattern

		Predicted									
		A	B	C	D	E	AD	DE	LD	FS	SUSP
Actual	A	91.7	5.2					1.3			2.1
	B	3.4	92.4		1.4		2.8				
	C	36.4		63.6							
	D		25		70		5				
	E	29.2	8.3			45.8					16.7
	AD		8.8				88.8	2.4			
	DE	6.2					9.2	83.1			1.5
	LD								100		
	FS	5.3								78.9	15.8
	SUSP									4.9	95.1

For each of the 10 classes (Class 1 has the A pattern, Class 2 has the B pattern, Class 3 has the C pattern, Class 4 has the D pattern, Class 5 has the E pattern, Class 6 has the AD pattern, Class 7 has the DE pattern, Class 8 has the LD pattern, Class 9 has the FS pattern, and Class 10 has the SUSP pattern), ROC (along with the AUC) curve and PR curve were assessed separately, as shown in Figure 4.4 and Figure 4.5, respectively.

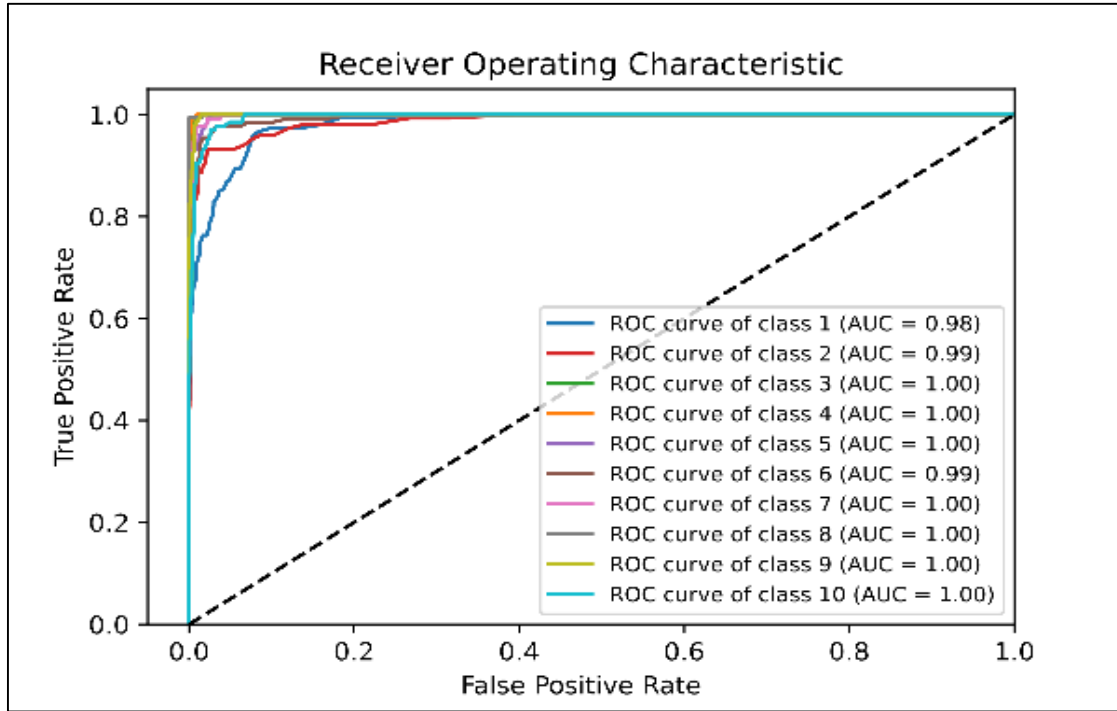


Figure 4.4. ROC Curve (AUC) of the Proposed CTG Morphological Pattern Model

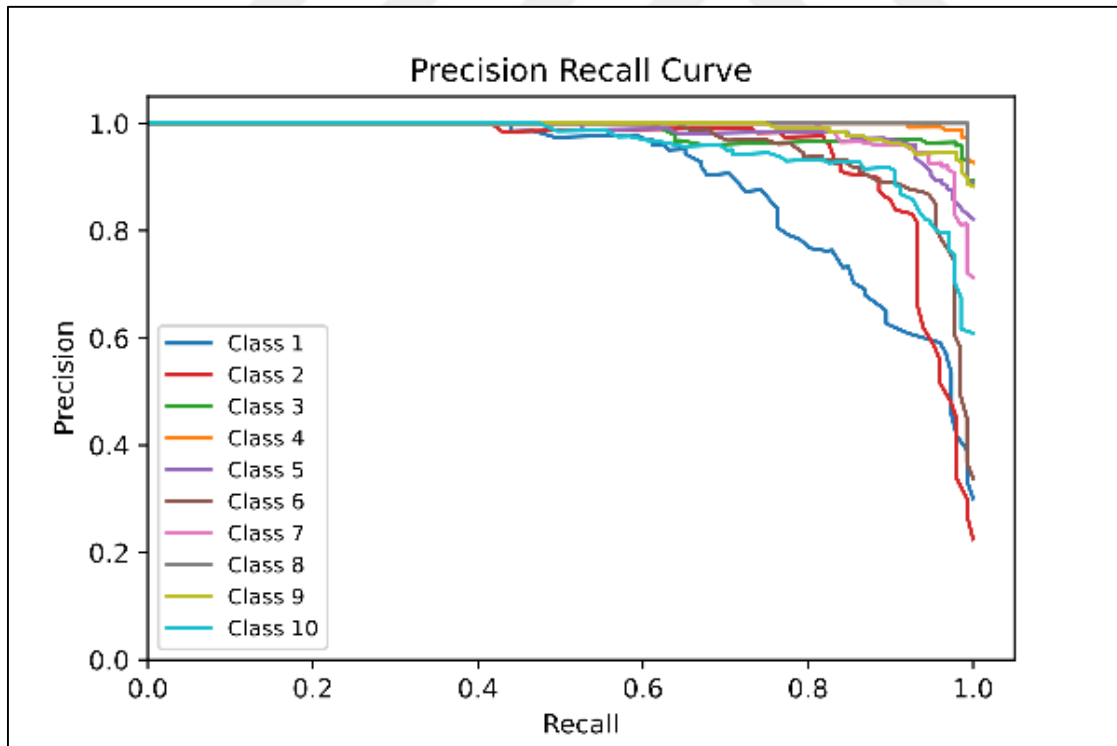


Figure 4.5. PR Curve of the Proposed CTG Morphological Pattern Model

Figure 4.6 shows the variance in model accuracy over the course of the complete iteration of suggested phase A model for CTG morphological pattern. When 14 features were selected then 94.96% was the maximum accuracy achieved by suggested phase A model.

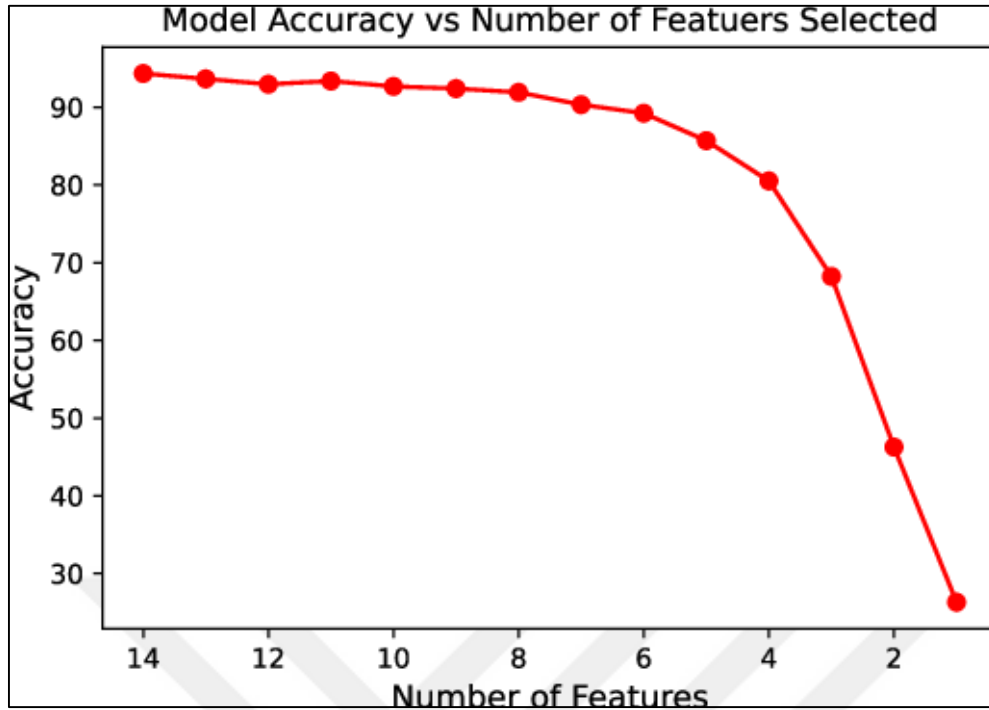


Figure 4.6. Relationship between Model Accuracy and the Total Number of Features Selected for CTG Morphological Pattern in Phase A

Full run for fetal status is shown in Figure 10 with three fetal conditions used as the intended output, while the full run for CTG morphological Pattern is displayed in Figure 13 with ten CTG classes (mentioned in Table 3) used as the intended output. As there were only three classes in the output for the fetal status part, it was simpler to categorize and classify that model. In contrast, the output for CTG morphological pattern model included ten classes, thus better classification was quite challenging to achieve. This is why there is a variation in the accuracies of the two graphs. Even still, the proposed model outperformed the scenario where only basic RF classifier (excluding the proposed model).

4.1.3. Bayesian optimization in Phase A

The major objectives of using BO in the proposed model are to more effectively observe the hyperparameter space, lower the computing resource while tweaking hyperparameters in such a manner that increased effectiveness of the suggested phase A model. With the part of fetus status, for example, the accuracy would be 96.54% if 13 features were selected (after FS module in the proposed algorithm) and no BO was applied. Although, the accuracy for 13 features is 96.62% if BO is used after the RFE module. In essence, the proposed model is improved upon using BO. The results of the proposed model while implementing BO are

displayed in Table 4.1 and in Table 4.3 whereas the results of the proposed model without implementing BO in are displayed Table 4.9 (showing suggested phase A model performance without BO) and Table 4.10 (showing confusion matrix of suggested phase A model without BO) allows one to observe the impact of BO on the proposed model of Phase A. The confusion matrix elements in Table 4.10 are in percentages to make comparisons easier.

Table 4.9. Performance Evaluation of the Proposed Fetal Status Model, without BO

Fetal Status	Precision	Recall	Specificity	F1-Score
Normal	0.9844	0.9313	0.9928	0.9584
Suspect	0.9205	0.9825	0.9595	0.9505
Pathological	0.9929	0.9792	0.9962	0.9860

Table 4.10. Confusion Matrix of the Proposed Fetal Status Model, without BO

		Predicted		
		Normal	Suspect	Pathological
Actual	Normal	93.4%	6.1%	0.5%
	Suspect	1.5%	98.3%	0.2%
	Pathological	0.0%	2.1%	97.9%

The use of BO in the proposed model had a significant influence on the classification performance of the model for pathological cases, in addition to a little (0.08%) improvement in accuracy of the model from 96.54% (no implementation of Bayesian optimization) to 96.62% (implementation of Bayesian optimization). Some results in these two tables were unchanged for normal and suspect cases. But the primary objective of using Bayesian optimization in suggested phase A model was the enhancement of estimation of pathological case. For example, there was a 33.33% drop in the wrong (or incorrect) identification of the pathological entries as the suspect entries and an increase in the right (or correct) identification of the pathological entries of roughly 0.71%. Even these minor advancements can result in extra lives being saved in a delicate field like fetal wellbeing.

Following the BO module, the optimal hyperparameters (using RF as a base classifier) of suggested phase A model for the fetal status as well as CTG morphological pattern are as follows:

For fetal status case the parameter of “estimator number” was 214, whereas, for CTG morphological pattern case it was 179. The split criterion for both types was entropy. Similarly, the max features to take into account during a split at a node for both the cases was square root of number of features. Lastly, class weights parameter for both cases was balanced mode. In the balanced mode, weights are automatically adjusted inversely proportionate to the class frequencies in input using the values of y (output).

4.1.4. Overview of SHAP analysis in Phase A

The SHAP summary plot is a Beeswarm-type figure, where y-axis represents predictors (or features) (which are often ranked according to their relevance), also x-axis represents SHAP values (which show participation of every predictor or feature to the anticipated target output). In both parts, the SHAP resulted in a different output because CTG morphological pattern had ten classes whereas fetal status had three. New Extracted Features (NEFs), that varied between NEF 1 till NEF 14, were assigned to new features when the dataset was fed to the autoencoder module. A blue dot represented a low feature value, whereas a red dot represented a high feature value. The color range between blue and red was shown for non-binary situations (such as those in this study), with purple representing the intermediate feature value. Every test set data point's individual SHAP value was represented by a dot. The length of the horizontal bars along the x-axis showed the magnitude of the impact of each feature on the model while the width of the bars reflected the range of SHAP values for each feature. Figure 4.7 shows NEF5 to be most positive influence on the phase A model in the fetal status part. The model is most negatively impacted by NEF4, NEF9, NEF6, NEF4, and NEF14.

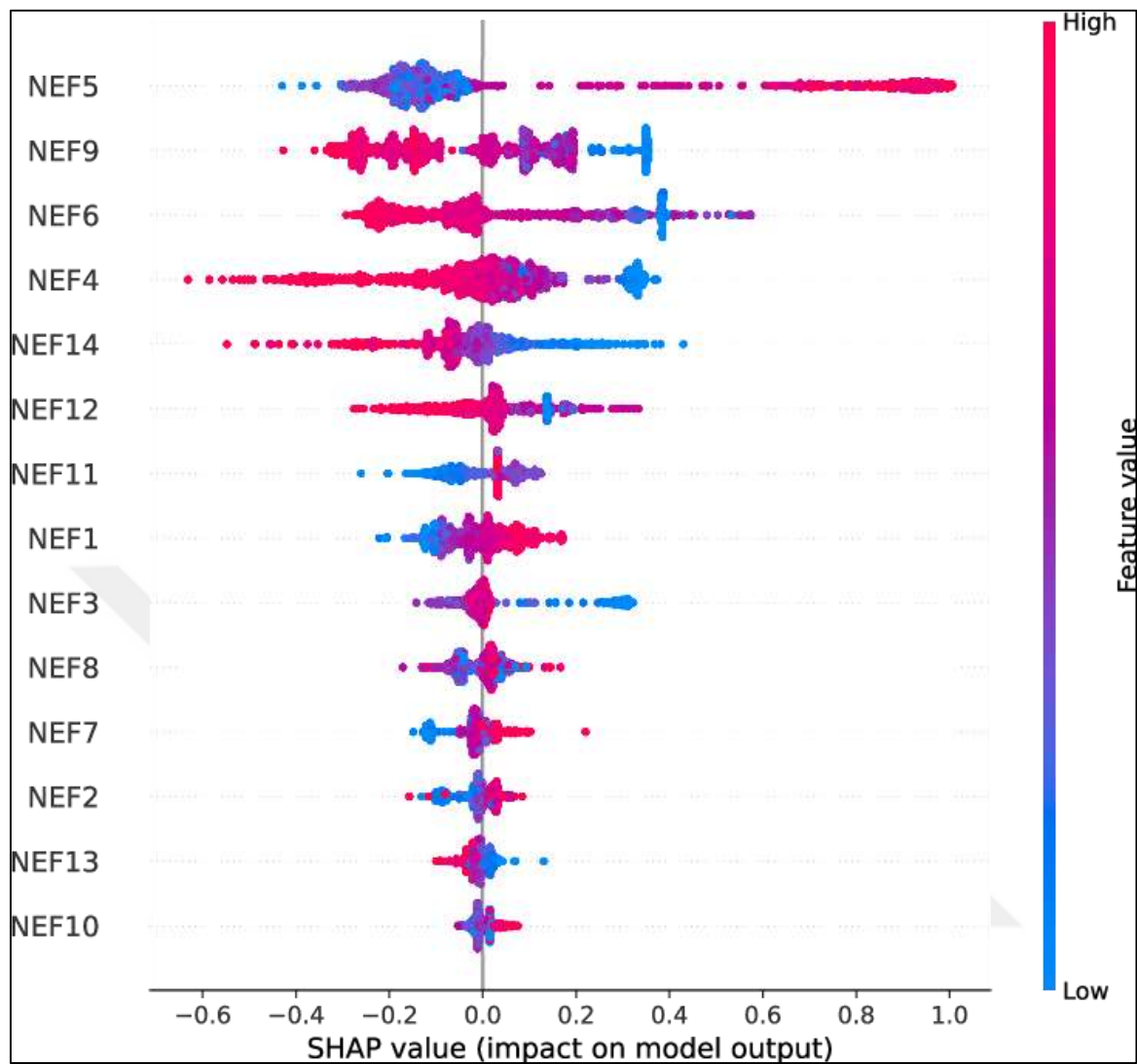


Figure 4.7. The Fetal Status SHAP Analysis Summary Plot

NEF14 has the most positive impact or effect on suggested phase A model for CTG morphological pattern part, as seen in Figure 4.8. However, the impact intensity ranges from low and medium (which is indicated with blue as well as purple dots, respectively). This model is most negatively impacted by NEF11. Whereas NEF4, NEF6, and NEF12 have a significant positive effect on suggested phase A model.

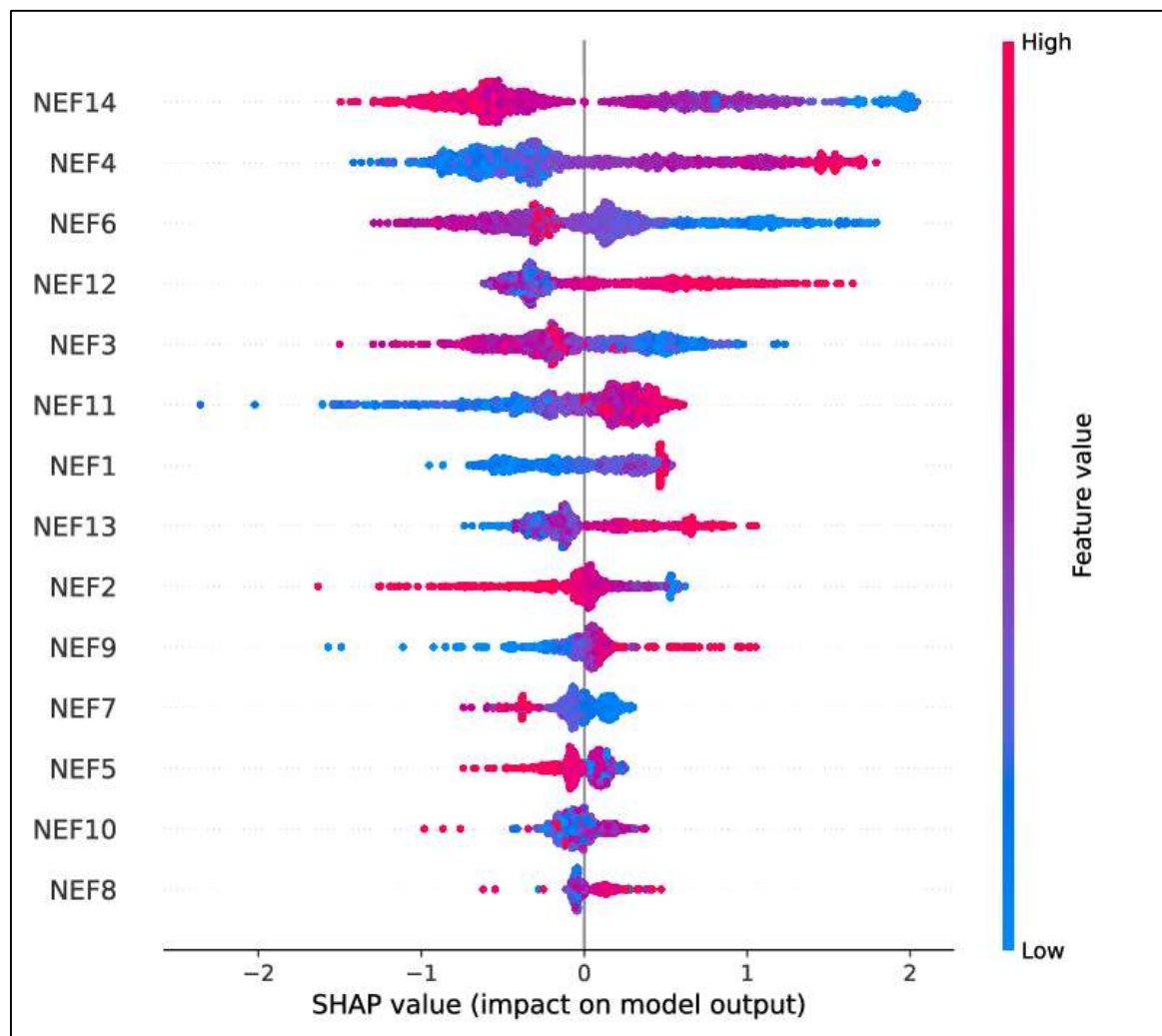


Figure 4.8. The CTG Morphological Pattern SHAP Analysis Summary Plot

The primary distinction between 2 SHAP analysis plots is that the NEFs imparted a large, positive as well as negative influence on the model (max effect was observed even at 1.0), and furthermore less predictors (or features) imparted a more meaningful impact, for the fetal status case. In contrast, NEFs had a significantly greater influence on the model (in CTG morphological pattern case) (maximum impact achieved around 2.0) as well as large number of predictors (or features) had an effect on the suggested phase A model. A study [55] related to the CTG dataset [19] also implemented SHAP analysis. Although, several of the starting predictors (like NZEROS - that was the number of Histogram zeros as well as DS- that were the severe decelerations in the CTG) imparted no effect at all on suggested phase A model as was observed from the results of [55]. All of those unnecessary features had been eliminated with the help of suggested phase A model. Had the same zero impact

features been used in this research then the model performance would have suffered. Hence, in the proposed model all new features had an impact.

4.2. Results of Phase B

Python 3.8 environment was used to perform the simulations in Phase B. For this phase, CTG dataset [19] was used. Three parts made up the simulations. Part 1 was the control experiment with all 21 features involved. Part 2 involved the simulation of the proposed mechanism (with feature elimination and hyperparameter optimization). In Part 3, hyperparameter optimization was performed on the proposed mechanism (via the best classifier found in Part 2). For parts 1 and 2, ROC (with AUC) was calculated separately for each of the three classes (where Class 1 had the Normal cases, Class 2 had the Suspect cases, and Class 3 had the Pathologic cases). For Phase B, the CLASS output (CTG morphological pattern) was not used.

4.2.1. Control Setup in Phase B

In this part 1, different classifiers were selected based on the literature review. These classifiers were KNN, MLP, SVC, DT and RF. Initially 75:25 was set as the training: testing ratio. Since all 21 features were used in training the models, this part served as the control. The proposed model for Phase A was not implemented in this part. As KNN, MLP, DT and RF can naturally support multiclass classification, so for this phase they posed no issue. However, for SVC, a slight modification was implemented. For the decision function shape in SVC, one-vs-one (“ovo”) is used as this is a multi-class case. The selection of ovo instead of ovr (one-vs-rest) was based on the results of the optimization. The comparison between the performance (based on the performance metrics of Table 3.1) of all these classifiers is given in Table 4.11.

Table 4.11. Performance Comparison of Classifiers in Part 1 (all 21 features)

Classifier	Fetal Status	Precision	Recall	F1-Score	Accuracy
KNN	Normal	0.92	0.98	0.95	89.66 %
	Suspect	0.73	0.57	0.64	
	Pathological	0.88	0.70	0.78	
MLP	Normal	0.94	0.97	0.96	91.54 %
	Suspect	0.77	0.67	0.72	
	Pathological	0.89	0.80	0.84	
SVC	Normal	0.93	0.97	0.95	90.60 %
	Suspect	0.71	0.65	0.68	
	Pathological	0.93	0.78	0.85	
DT	Normal	0.94	0.96	0.95	90.98 %
	Suspect	0.74	0.64	0.69	
	Pathological	0.90	0.86	0.88	
RF	Normal	0.95	0.98	0.97	93.61 %
	Suspect	0.83	0.69	0.76	
	Pathological	0.92	0.90	0.91	

The confusion matrix of the best classifier in Part 1, Random Forest, is shown in Table 4.12. The Random Forest classifier predicted normal and pathologic cases with good confidence margins (above 90% for both). However, for the suspect cases, it incorrectly predicted 26.4% of suspect cases as normal cases, which is an unacceptable figure from a medical standpoint. This incorrect prediction needs to be improved (addressed in part 2). The entries in Table 4.12 are shown in percentages for ease of comparison.

Table 4.12. Confusion Matrix of The Best Classifier (RF) in Part 1 (all 21 features)

		Predicted		
		Normal	Suspect	Pathological
Actual	Normal	98.3%	1.5%	0.2%
	Suspect	26.4%	69.4%	4.2%
	Pathological	2 %	8 %	90 %

4.2.2. Feature Elimination in Phase B

The part 2 of Phase B had further three sub-parts. In Part 2A, feature importance was calculated. There are many methods to find the importance of features but for this research, GINI importance of features via RF and importance of features via XGBoost were used to rank the features, based on their respective importance. The importance of features for both XGBoost and RF are shown in Figure 4.9 and Figure 4.10 respectively. The importance of each feature was found using two different techniques to avoid over-exaggerating the

importance of features and to provide a comparative angle in this research. In the figure, it can be observed that both techniques produced different importance of features per feature. However, the least important four features that are approximately common in both graphs are DL, DS, NZEROS, Tendency. They were eliminated from the dataset. The choice for the number of features to eliminate was based on both the literature review as well as trial and error. As the experiment progressed and more observations were made, more features would be eliminated. Eventually, the dataset was reduced to 17 features, which would be used for fetal status classification.

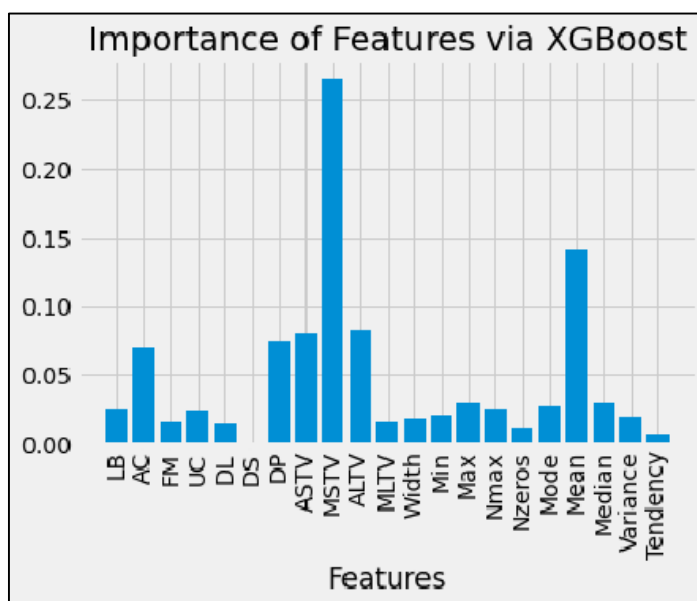


Figure 4.9. Importance of Features in CTG Dataset via XGBoost

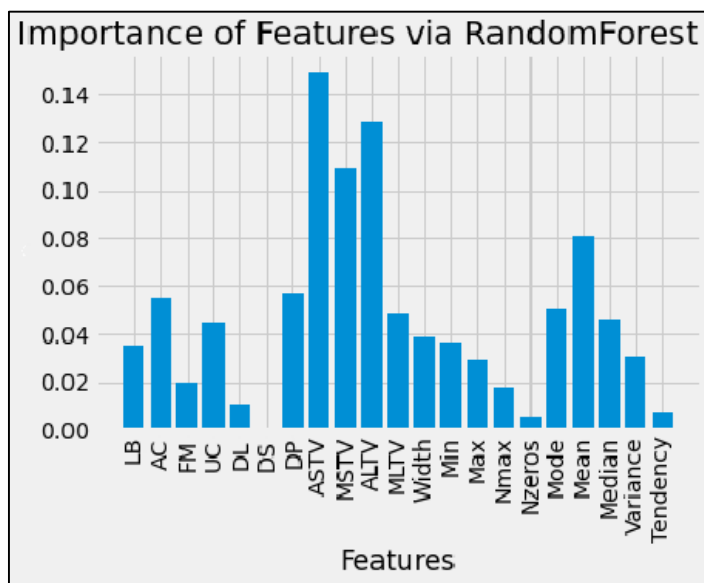


Figure 4.10. Importance of Features in CTG Dataset via RF

The optimal training-to-testing ratio of 95:05 was used here for all the classifiers; the reason for the selection of this ratio is explained in detail in Part 2C. The remaining 17 features found via the proposed feature elimination method (in Part 2A) were used in Part 2B. In this part the selected hyperparameters to fine tune were as follows:

- For KNN: Number of neighbors, Weight function, Algorithm to compute the nearest neighbor, Power parameter for the Minkowski metric.
- For MLP: Hidden layer size, Learning rate, Activation function, Weight optimization solver, Regularization term (L2 penalty) (alpha).
- For SVC: Regularization parameter (C), Kernel coefficient (gamma), Class weights, Kernel.
- For DT: The function to measure the quality of a split (split criterion), The strategy used to choose the split at each node (splitter), Max number of features to look for a split.
- For RF: Tree amount in forest (estimators), Criterion for split, Max features to look for split, Class weights.

The classifiers performed better in this part, as shown in Table 4.13. The confusion matrix of the best classifier in Part 1, Random Forest, is shown in Table 4.14.

Table 4.13. Performance Comparison of Classifiers in Part 2B (17 features)

Classifier	Fetal Status	Precision	Recall	F1-Score	Accuracy
KNN	Normal	0.95	0.98	0.96	93.44
	Suspect	0.83	0.79	0.81	
	Pathological	1.00	0.75	0.86	
MLP	Normal	0.99	0.94	0.96	93.49
	Suspect	0.75	0.95	0.84	
	Pathological	1.00	0.75	0.86	
SVC	Normal	0.99	0.93	0.96	93.46
	Suspect	0.78	0.95	0.86	
	Pathological	0.80	1.00	0.89	
DT	Normal	0.98	0.96	0.97	95.33
	Suspect	0.89	0.89	0.89	
	Pathological	0.80	1.00	0.89	
RF	Normal	0.97	0.99	0.98	96.26
	Suspect	0.94	0.84	0.89	
	Pathological	1.00	1.00	1.00	

Table 4.14. Confusion Matrix of The Best Classifier (RF) in Part 2B (17 features)

		Predicted		
		Normal	Suspect	Pathological
Actual	Normal	98.8%	1.2%	0%
	Suspect	15.8%	84.2%	0%
	Pathological	0 %	0 %	100 %

The proposed feature elimination and hyperparameter optimization addressed the issue where a large number of Suspect cases or entries (in the dataset) had been predicted (incorrectly) as Normal cases. The proposed method reduced these incorrect predictions by 40.1% (from 26.4% to 15.8%). Moreover, it predicted the Pathologic cases with full confidence.

The ROC and AUC of the best classifier (Random Forest) from both Part 1 (with all 21 features) and Part 2B (with 17 features), are given in Figure 4.11 and Figure 4.12 respectively.

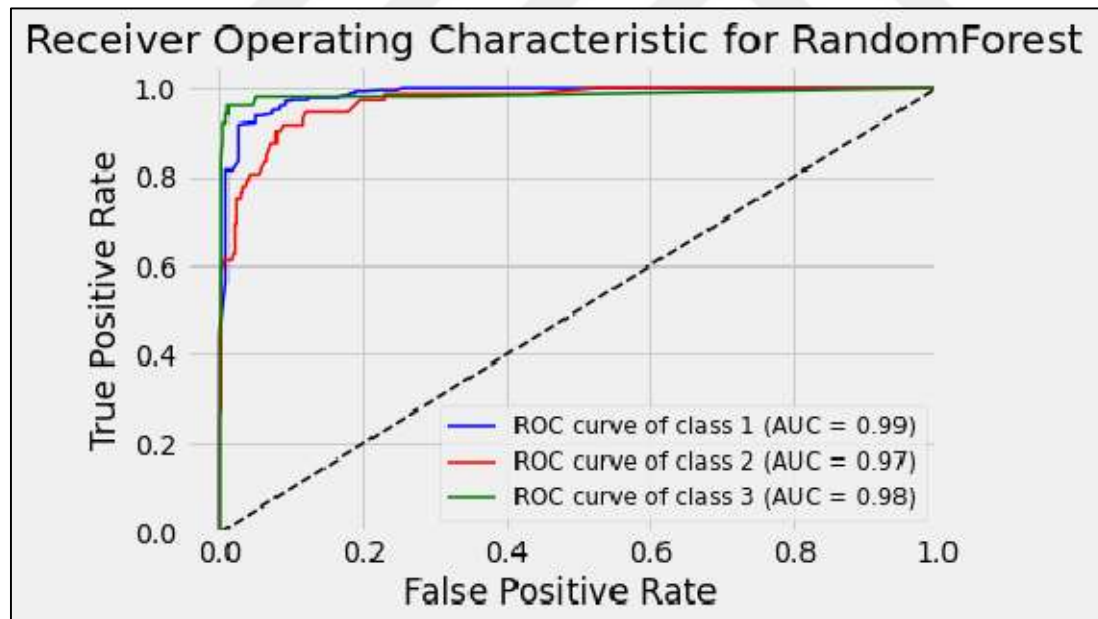


Figure 4.11. ROC Curve (AUC) of RF in Part 1 (with 21 features)

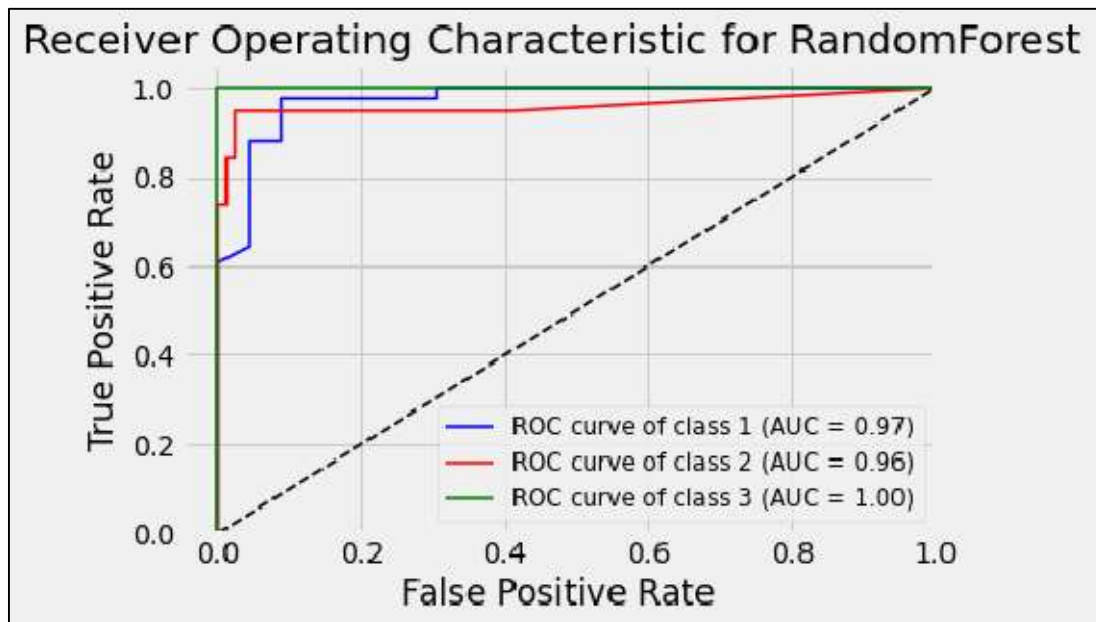


Figure 4.12. ROC Curve (AUC) of RF in Part 2 (with 17 features)

Since the dataset was relatively small compared to other medical datasets, the model was not getting enough training samples to make good predictions. In Part 2C, the number of training samples was increased in an incremental manner to observe the effect on the performance of the classifiers. From Figure 4.13, it was observed that increasing the training sample size increased the classifier accuracy for this particular dataset [19]. Thus, to get better results while also leaving enough samples for testing, the training-to-testing ratio of 95:05 was used in Part 2B. In part 2 only static feature elimination was performed. In part 3 however, dynamic feature elimination would be performed.

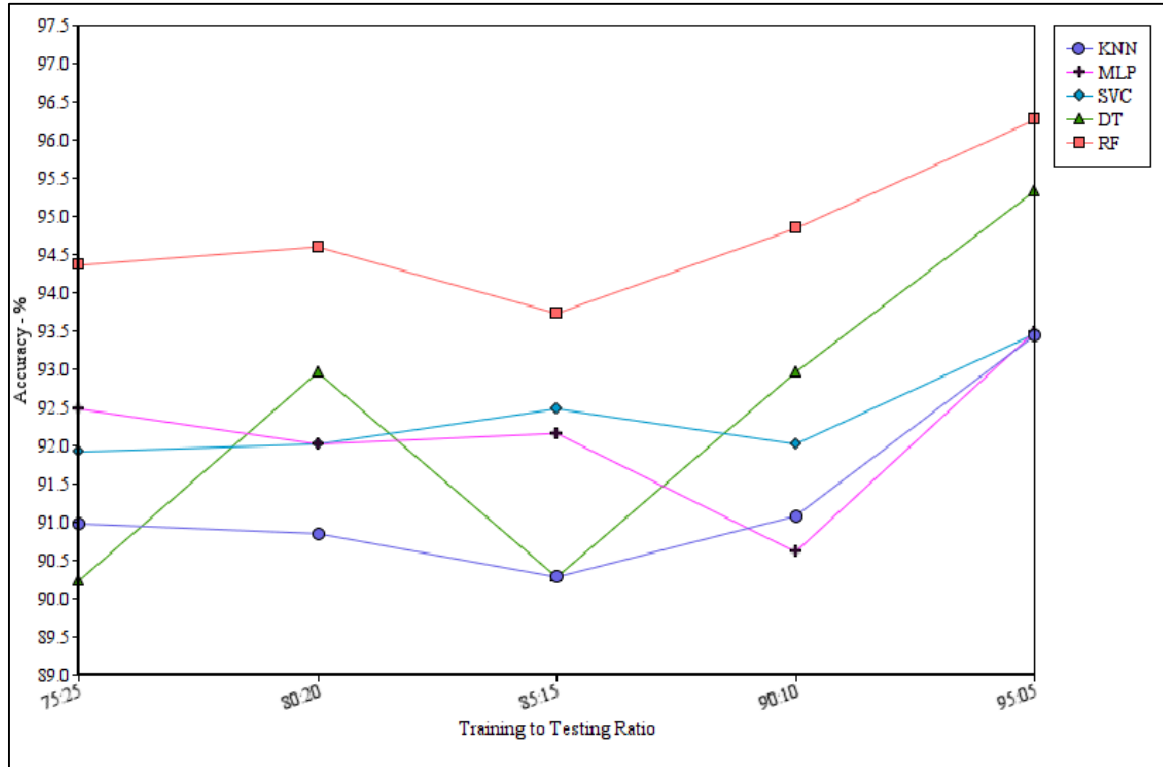


Figure 4.13. Effect of Training Sample Size on Model Accuracy

4.2.3. Hyperparameter optimization in Phase B

In this part 3, hyperparameter optimization was performed on the proposed mechanism (via the best classifier found in Part 2). From the results of part 2, it was observed that Random Forest was the best classifier for this CTG dataset [19]. This whole process was repeated for multiple training to testing ratios. In this research the training to testing ratios of 75:25, 80:20, 85:15, 90:10, 95:05 were used to get the optimum results. For hyperparameter optimization BO was used. While applying BO on the proposed model, further 6 features (FM, Variance, MAX, NMAX, MIN, LB) were automatically removed, resulting in a total number of 10 eliminated features. The performance analysis and confusion matrix of the proposed model is given in Table 4.15 and Table 4.16 respectively. With the proposed model, the overall accuracy was increased, and wrong estimation of Suspect entries was reduced by 60.22 %.

Table 4.15. Performance Analysis of The Proposed Model in Phase B (11 features)

Classifier	Fetal Status	Precision	Recall	F1-Score	Accuracy
RF	Normal	0.98	0.99	0.98	97.20
	Suspect	0.94	0.89	0.92	
	Pathological	1.00	1.00	1.00	

Table 4.16. Confusion Matrix of The Proposed Model in Phase B (11 features)

		Predicted		
		Normal	Suspect	Pathological
Actual	Normal	98.8%	1.2%	0%
	Suspect	10.5%	89.5%	0%
	Pathological	0 %	0 %	100 %

4.3. Results of Phase C

Python 3.8 platform was used to perform the related simulations of suggest phase C model. The ratio of 80:20 was used as the ratio of training to testing. Artificial FECG data signals were trimmed around 3000 samples. This was performed to achieve an easily readable display, and so that detecting an anomaly could be displayed easily in the figures. From the simulation type discussed in section 3.4 the following cases were used in this study: Case 1, Case 2, and Case 4. Reason for selecting only these cases from all the cases in the FECGSYN [20] dataset was that these cases primarily focused on fetal distress. Since one of main objective of this research study was to track and observe the fetal movements (which can also be caused due to fetal distress).

4.3.1 FECGSYN Case 1 of Phase C

This case simulated the fetal activity with noise added. For this case, the simulated FECG signal is shown in Figure 4.14. The simulated signal was used to train LSTM suggested phase C model. Moreover, model loss (based on training loss as well as validation loss) is shown in Figure 4.15.

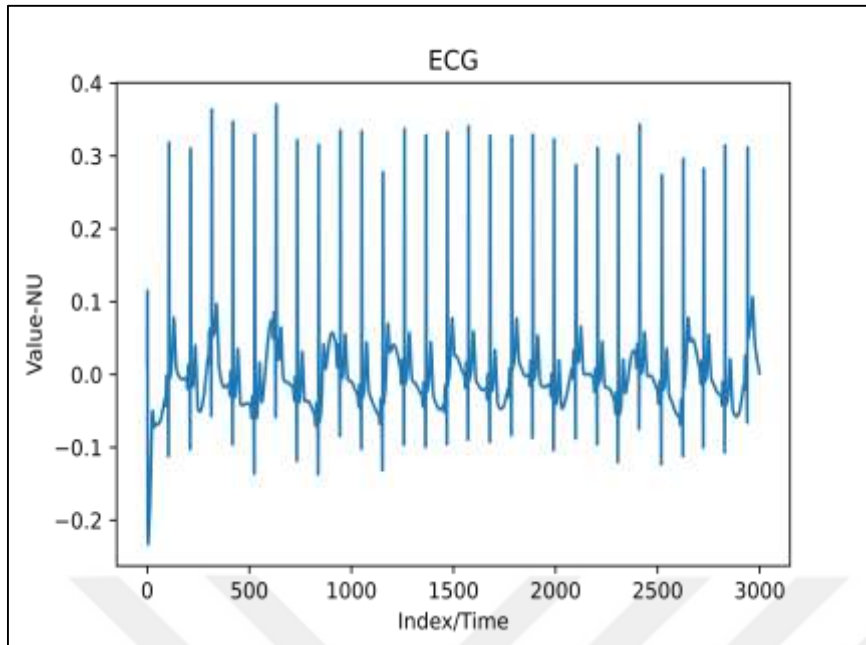


Figure 4.14. Case 1 FECGSYN Signal

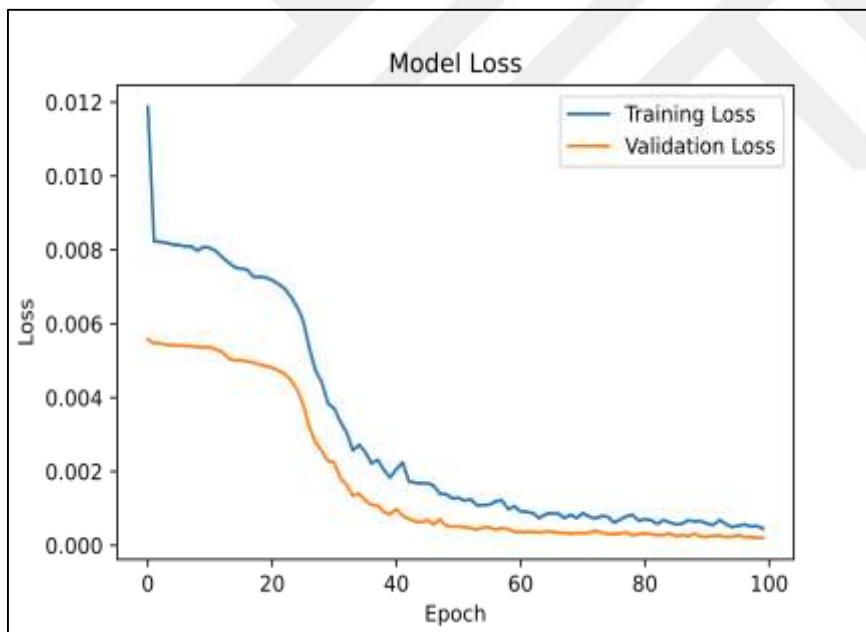


Figure 4.15. Case 1 Model Loss

The proposed model based on LSTM (case 1), estimated simulated FECG signals accurately as displayed in Figure 4.16.

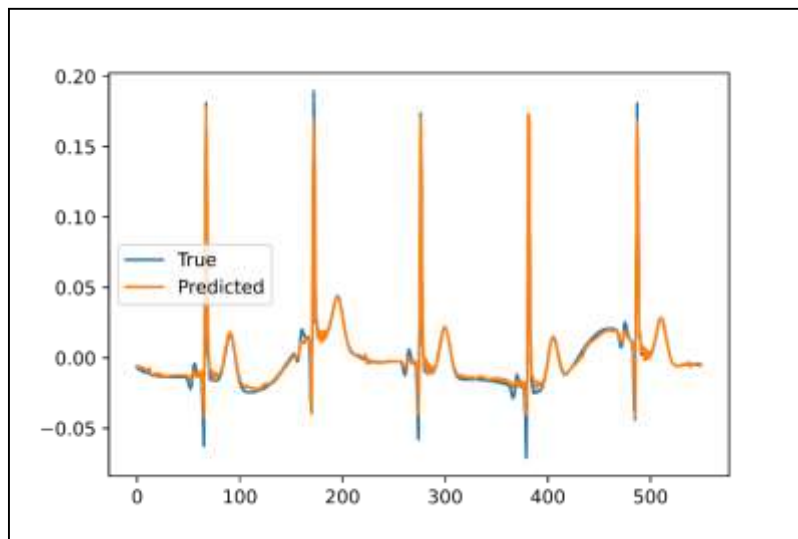


Figure 4.16. Case 1 Predicted FECG Signal

With the given four SNR variations (03, 06, 09, and 12 dB) anomaly identification was implemented as displayed in Figure 4.17 (for case 1). It is shown that added noise in signal used for this case 1 (Figure 4.17) resulted in many alterations in signal. By changing the threshold these alterations can be observed. Such that smaller threshold results in the identification of a larger number of anomalies.

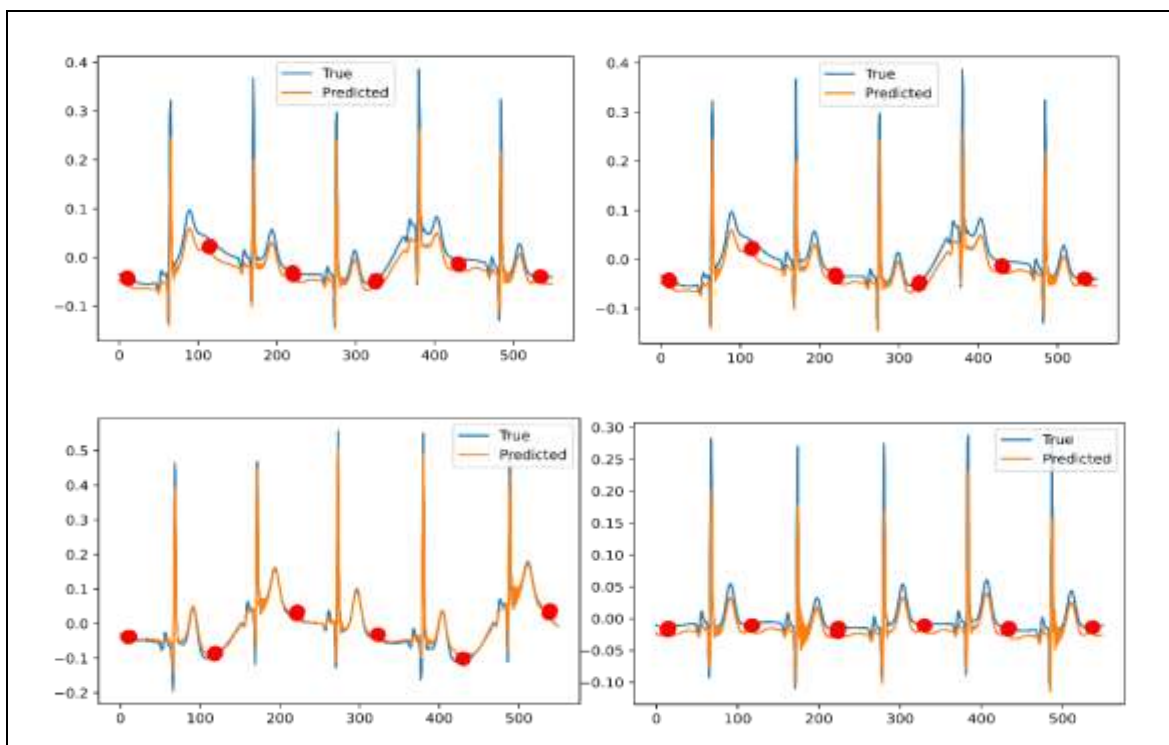


Figure 4.17. Case 1 Anomaly Detection via Proposed Model. Top Left to Right: 03 dB SNR, 06 dB SNR. Bottom Left to Right: 09 dB SNR, 12 dB SNR

4.3.2. FECGSYN Case 2 of Phase C

This case simulated the accelerations or decelerations of the fetal heart rate with added noise. For this case, the simulated FECG signal is shown in Figure 4.18. The simulated signal was used to train LSTM suggested phase C model. Moreover, model loss (based on training loss as well as validation loss) is shown in Figure 4.19.

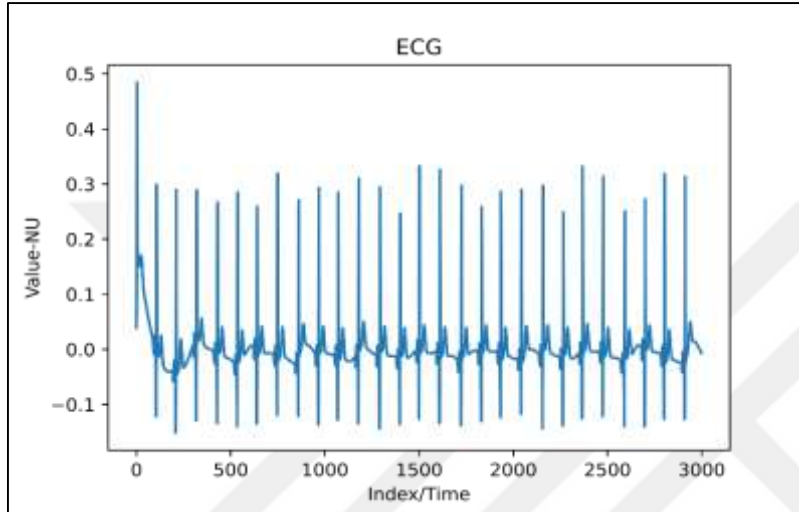


Figure 4.18. Case 2 FECGSYN Signal

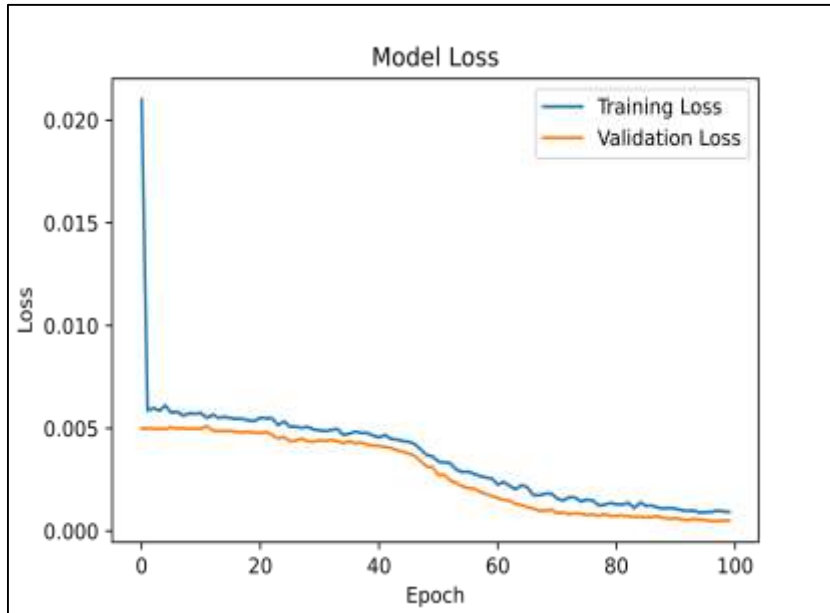


Figure 4.19. Case 2 Model Loss

The proposed model based on LSTM (case 2), estimated simulated FECG signals accurately as displayed in Figure 4.20.

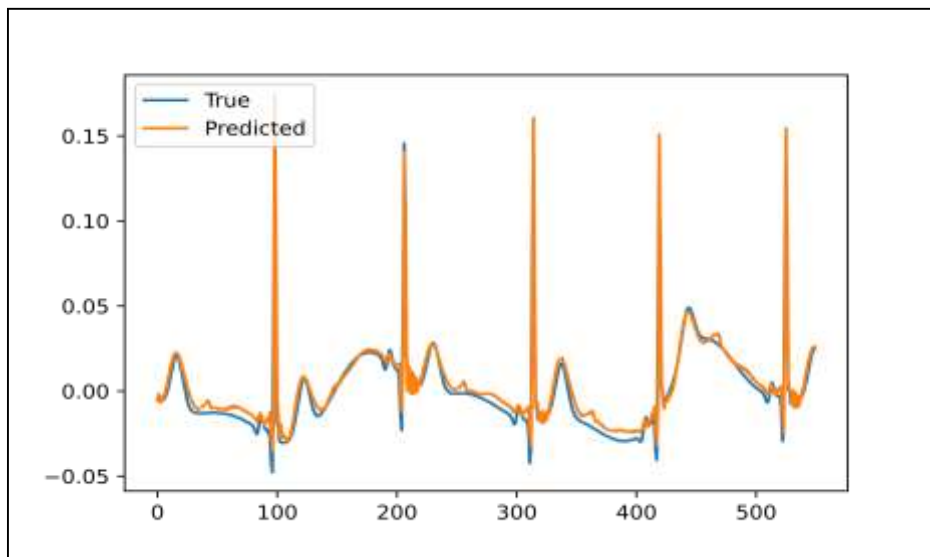


Figure 4.20. Case 2 Predicted FECG Signal

With the given four SNR variations (03, 06, 09, and 12 dB) anomaly identification was implemented as displayed in Figure 4.21 (for case 2). It is shown that added noise in signal used for this case 2 (Figure 4.21) resulted in many alterations in signal. By changing the threshold these alterations can be observed. Such that smaller threshold results in the identification of a larger number of anomalies.

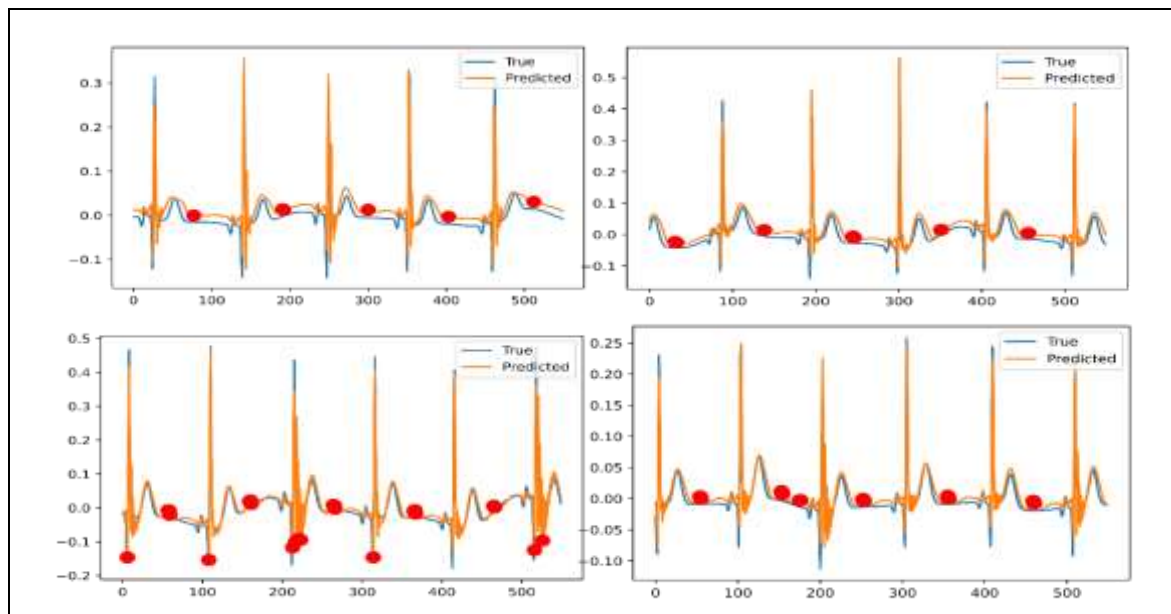


Figure 4.21. Case 2 Anomaly Detection via Proposed Model. Top Left to Right: 03 dB SNR, 06 dB SNR. Bottom Left to Right: 09 dB SNR, 12 dB SNR

4.3.3. FECGSYN case 4 of Phase C

This case simulated the fetal ectopic beats with added noise. If a heart does not beat normally then such variations in the heart beats are called ectopic beats. In the FECG plot, ectopic beats result in skipped or extra beats. For this case, the simulated FECG signal is shown in Figure 4.22. The simulated signal was used to train LSTM suggested phase C model. Moreover, model loss (based on training loss as well as validation loss) is shown in Figure 4.23.

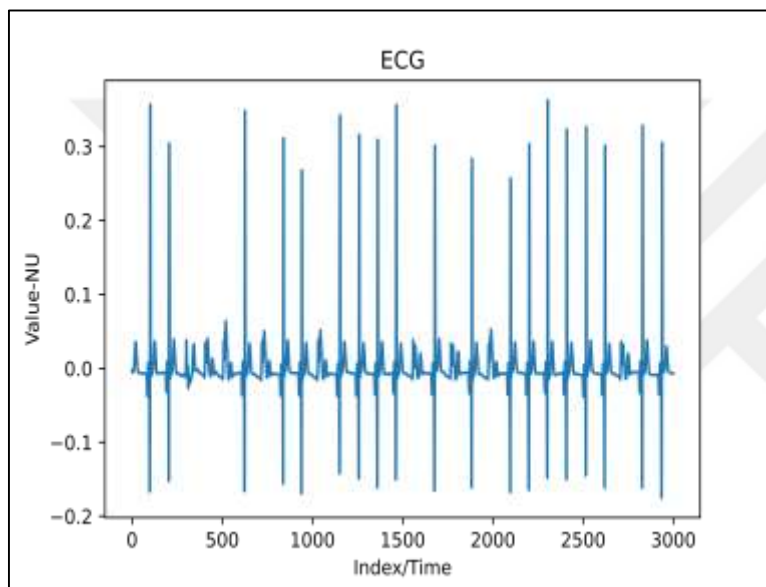


Figure 4.22. Case 4 FECGSYN Signal

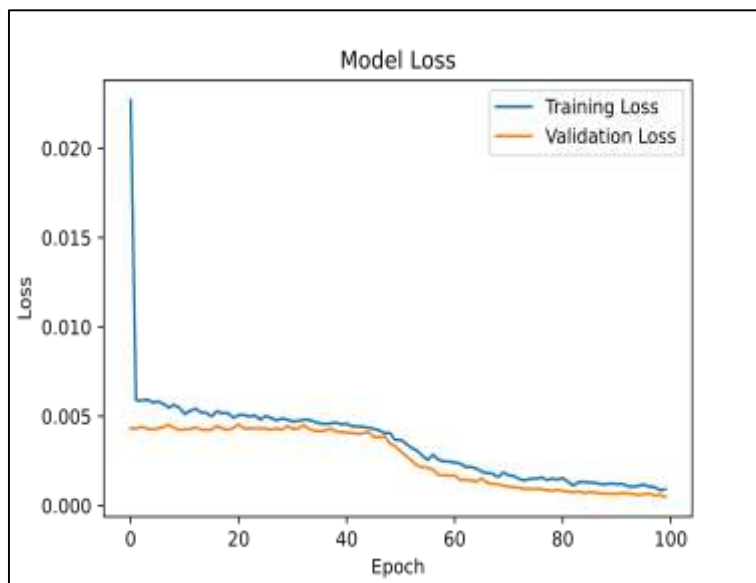


Figure 4.23. Case 4 Model Loss

The proposed model based on LSTM (case 4), estimated simulated FECG signals accurately as displayed in Figure 4.24.

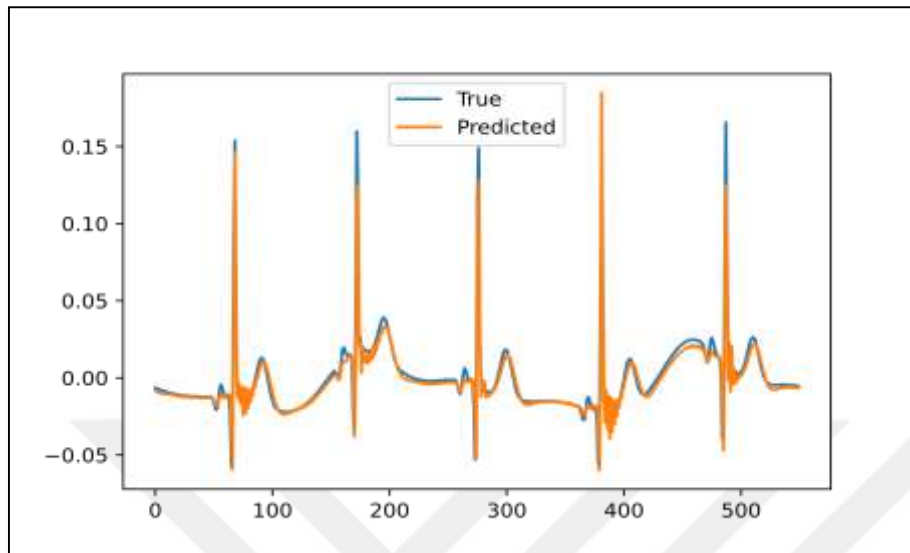


Figure 4.24. Case 4 Predicted FECG Signal

With the given four SNR variations (03, 06, 09, and 12 dB) anomaly identification was implemented as displayed in Figure 4.25 (for case 4). It is shown that added noise in signal used for this case 4 (Figure 4.25) resulted in many alterations in signal. By changing the threshold these alterations can be observed. Such that smaller threshold results in the identification of a larger number of anomalies. Threshold value was knowingly set at a small value. This was done because of the fact that the number of anomalies is high as there are missed beats in this case (as seen in Figure 4.22). Thus, this also allowed the proposed phase C model to identify more anomalies in this case.

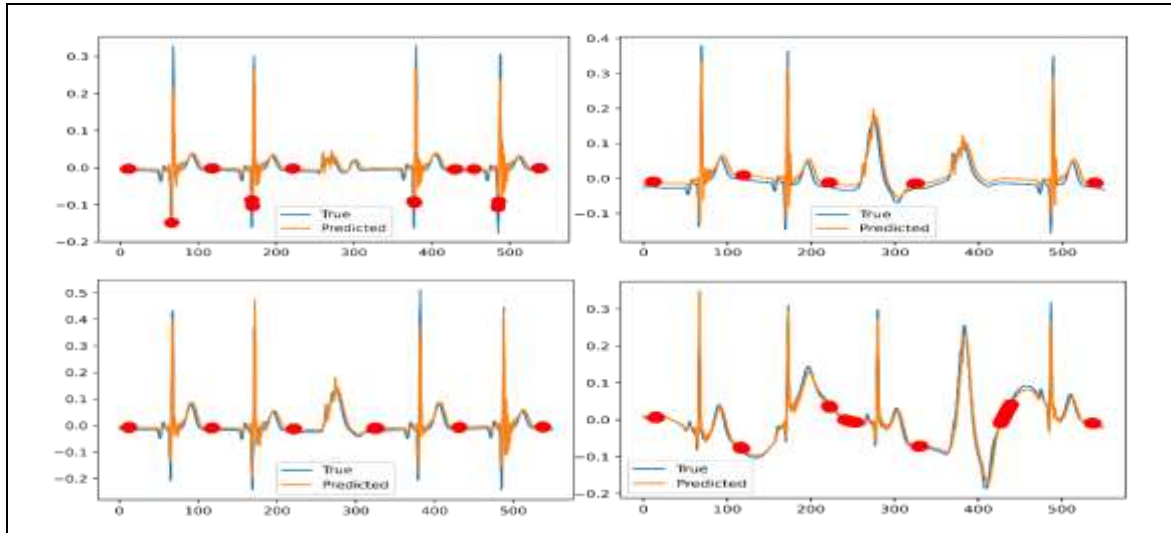


Figure 4.25. Case 4 Anomaly Detection via Proposed Model. Top Left to Right: 03 dB SNR, 06 dB SNR. Bottom Left to Right: 09 dB SNR, 12 dB SNR

4.3.4. Performance Analysis of Phase C

For all the cases of FECGSYN used in this phase C, their performance indicators as well as detection of anomalies are displayed in Table 4.17. In table C1 is the dataset code for Case 1 (Fetal movement with noise), C2 is the dataset code for Case 2 (FHR accelerations/decelerations with noise), and C4 is the dataset code for Case 4 (Ectopic beats for fetus with noise). Based on different values of threshold different number of anomalies were detected in the FECG signals. The table also displays different versions of the same case having different SNR values (of 00 dB, 03 dB, 06 dB, 09 dB and 12 dB). This was done to increase the quality of the performance analysis of the proposed model in Phase C. Also, to check whether increasing noise in the signal can affect the proposed model.

Table 4.17. Performance Indicators and Threshold Relation with Anomalies

Dataset Case	SNR	MSE	RMSE	Threshold	Anomalies Detected
C1	00dB	0.0002	0.0143	0.5	6
	03dB	0.0007	0.0273	0.5	10
	06dB	0.0012	0.0352	0.5	12
	09dB	0.0040	0.0195	0.5	10
	12dB	0.0011	0.0331	0.5	10

Table 4.17. (continue) Performance Indicators and Threshold Relation with Detected Anomalies

C2	00dB	0.0005	0.0229	0.5	8
	03dB	0.0021	0.0455	0.3	6
	06dB	0.0015	0.0389	0.3	7
	09dB	0.0009	0.0312	0.35	23
	12dB	0.0007	0.0258	0.21	98
C4	00dB	0.0005	0.0220	0.5	12
	03dB	0.0010	0.0321	0.16	21
	06dB	0.0007	0.0266	0.16	22
	09dB	0.0010	0.0321	0.24	15
	12dB	0.0007	0.0258	0.4	41

5. DISCUSSION

For Phase A, the overall trend, as seen in Figure 4.3 and Figure 4.6, shows a negative correlation between the number of features and the proposed model accuracy, with fewer features resulting in poorer model accuracy. The proposed model performance analysis metrics varied from 0.92 to 0.99 as seen in Table 4.1. This is a good range for a model. In comparison to when the Random Forest (without proposed mechanism) was implemented on dataset of CTG [19], this is a huge improvement. The recall and precision entries of suspect cases were extremely small (0.69 for recall and 0.83 for precision) for scenario when only basic RF was utilized (excluding the proposed model), however in suggested phase A model, those entries were comparatively better as, 0.98 for recall and 0.92 for precision.

As observed in Table 4.1 suggested phase A confusion matrix for fetal status case, a significant decrease was observed in erroneous estimations of suspect case and pathological case was made. When standard Random Forest (excluding suggested phase A model) was applied, suspicious entries were mistakenly forecasted as normal cases by 26.4%. However, utilizing suggested phase A model, this erroneous estimation was reduced till just 1.5%, that resulted in lowering of 94.31% in false positives for suspect and normal cases. This reduction in inaccurate and incorrect prediction by the proposed model is a huge advantage for this model, especially for a delicate topic like fetal wellbeing. The overall model accuracy was 96.62% for the fetal status part.

The ROC (as seen in Figure 4.1) and PR (as seen in Figure 4.2) curves for the fetal status part give useful information regarding the proposed model capacity to predict confidently and reliably each of the three classes. Moreover, all AUC values are over 0.99. The proposed model performs exceptionally well in categorizing as well as estimating the pathological case, which is most significant finding from the ROC and PR curves. As a pathological case is more worrisome than a normal case in a medical setting. This is due to the fact that pathological cases need prompt attention, allowing the fetal wellbeing to be restored.

For normal cases, standard Random Forest (excluding suggested phase A model) resulted in good observations, but suspicious case and pathological case were interestingly not estimated with a high degree of confidence. Numerous suspicious cases were misclassified as normal cases or pathologic cases. When normal cases are erroneously estimated as

suspicious cases or pathological cases, then this erroneous estimation offers many issues due to the medical ramifications. The precise categorization and classification of suspicious case and pathological case in classification of fetal status is more important than normal case classification. As a result, suggested phase A model was successful in raising the margins of confidence in identifying suspicious cases as well as pathological cases.

For morphological pattern of CTG part of Phase A, model accuracy was 94.96%. Model accuracy was 87.22% when standard Random Forest (excluding suggested phase A model) was utilized, therefore this represented an increase of 8.87%. In that case class A, class B, as well as class LD resulted in significantly more accurate estimations for the standard RF classifier in comparison to the other cases of morphological pattern of CTG part, shown in Table 4.3. Additionally, simply utilizing standard Random Forest (excluding suggested phase A model), the correct predictions for E class (which is the switch class between Calm Sleep, and Suspect pattern) were amounting to 45.8% only (in this case the recall was 0.45). But suggest phase A model greatly decreased erroneous estimations across all output classes. The F1-score also showed improvement, with all morphological patterns displaying favorable results of the performance metrics. The F1-score for every class (apart from class A) was more than 0.91. An analysis of the confusion matrix (as seen in Table 4.4) may also be used to show good workings of suggested phase A model. As an example, it can be observed that class E saw an improvement in the right estimation to 98.7%, or that also equates to an increase of 115.5%. Additionally, class E recall climbed to 0.98. Furthermore, when a comparison of utilizing standard Random Forest (excluding suggested phase A model), the proposed model has relatively less inaccurate predictions for the suspect and pathological linked classes (like FS class and SUSP class). While standard Random Forest (excluding suggested phase A model) was used, the mean percentage of correctly predicted CTG morphological patterns was 80.94%. While the average accuracy of predictions made using the proposed model across all classes was 94.99%.

The relation of fetal status categories and morphological pattern of CTG categories must be explored before going over ROC (along with ACU) curve as well as PR curve of suggested phase A model for morphological pattern of CTG. Fetal state categories are spread among all of morphological pattern of CTG categories, as can be seen in Figure 5.1. A class, B class, C class, and D class represent a fetal category that is normal. The AD class along with the DE class correspond to a category of fetus that is primarily normal with a small number of

suspect cases. The suspect case corresponds to Classes SUSP and E. The sleeping pattern of Class E alternates between a regular, peaceful slumber versus suspicious routine. The LD class along with the FS class also correspond to categories of pathological nature. A class is the sole class that experiences decline in both recall value as well as F1 score, despite suggested phase A model of morphological pattern of CTG having superior ROC curve (Figure 4.4) and PR curve (Figure 4.5) than basic RF (excluding proposed model). The majority of the inaccurate estimations for the A class were spread in other categories of normal nature linked morphological patterns of CTG, so there is some compromise here. Performance analysis metrics for the suspect case linked morphological patterns of CTG (like E class as well as SUSP class) along with the pathological case linked morphological patterns of CTG (like LD class as well as FS class) were excellent.

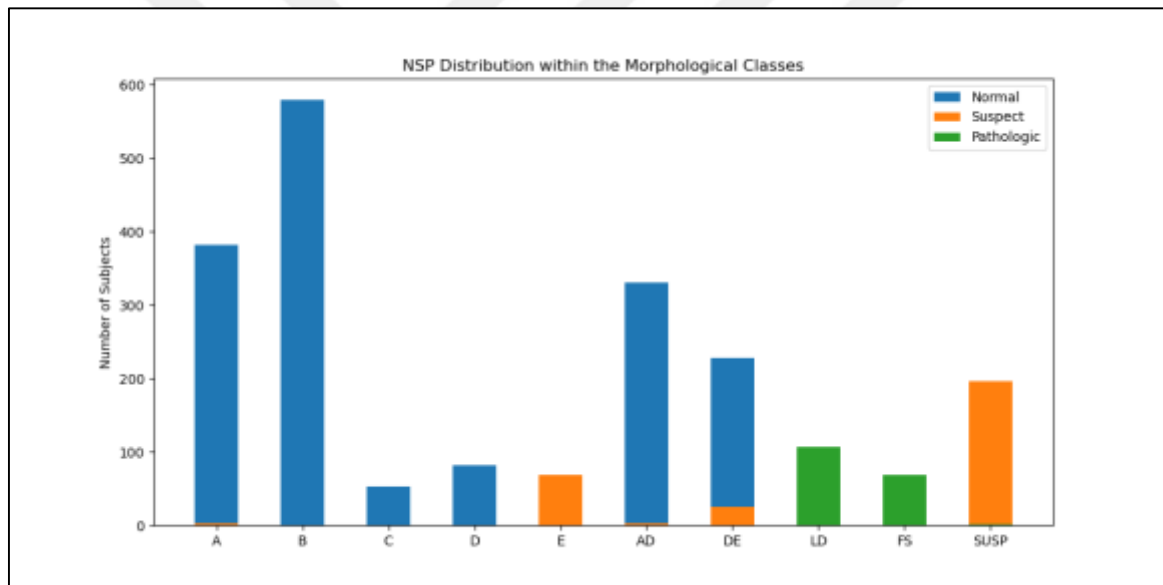


Figure 5.1. NSP Disposition in the CLASS of Morphological Pattern

An objective of this study is to make it simpler for future researchers using machine learning in cardiotocography while also choosing the adjusted hyperparameters from this work. Any future work related to this field would benefit from the knowledge provided in this research.

The fact that this CTG dataset [19] was compiled from participants in a developed countries is a significant drawback to using it at the global level because it ignores the developing and under-developed countries. Furthermore, this CTG dataset does not include information about the individual participants' social, demographic, or medical features (such as information on prenatal nutrition, maternal health, etc.). All of these factors have an impact

on the events in the third trimester and may be utilized to improve the proposed model of Phase A. To confirm the real global performance of the proposed model, similar CTG dataset should be taken from developing and under-developed countries so that the proposed model is able to learn the global CTG trends as well. Future related research may be performed towards the application of this model with hardware rather than just depending on the fixed CTG database. After the hardware integration, the proposed model would be able to perform clinical application of this model on CTG categorization. Future efforts might combine more classifiers to further increase the accuracy of the proposed model. The dataset size in this study was artificially expanded using SMOTE, and better findings were obtained. However, a further improvement can be made if more genuine entries are included in this CTG dataset [19]. A bigger and more current CTG dataset may be used in future work related to this study. Future work related to this proposed model may also involve the deployment across various stages of labor, as suggested by [57].

Suggested phase A model may be adapted so as to operate with other datasets of CTG which have been received via Sisporto applications because the current study employed a dataset of CTG which was taken from the version 2.0 of Sisporto. The 2015 version of FIGO recommendations for intrapartum fetal monitoring have also been incorporated into the most recent version of Sisporto, 4.0 [58]. Related study [59] also emphasizes the advantages of using computer based digital CTG (particularly Sisporto), this also highlights the fact that Sisporto has significant positive points over conventional CTG analysis in clinical practice. The idea that the use of Sisporto in medical treatment lowers the frequency of cesarean-based births is supported by a second study publication [60]. Consequently, Sisporto and the associated CTG dataset offer a useful resource in the field of CTG. The CTG dataset is often utilized in CTG-related experiments and research, as seen in Table 5.1 as well.

Table 5.1. Comparing this study (Phase A and Phase B) with relevant literature

Authors	Approach	Model Accuracy
Y. Zhang et al. [39]	SVM with AdaBoost	93%
M. Manikandan et al. [16]	RF with Bagging	96.61%
I. C. Stylios et al. [43]	KNN	73.33%
A. Batra et al. [25]	RF	93.41%
Y. Fei et al. [40]	FCM-ANFIS	96.39%
Z. Cömert et al. [31]	Resilient Backpropagation	93.60%
Z. Hoodbhoy et al. [27]	XGBoost	93%
N.J.A. Kadhim et al. [33]	Naïve Bayes classifier with Firefly algorithm	86.54%
K. Agrawal et al. [28]	DT	93.17%
A. K. Pradhan et al. [29]	RF	93%
M. Ramla et al. [30]	CART	90.12%
Z. Cömert et al. [44]	Extreme Learning Machine	93.42%
N. Chamidah et al. [22]	KNN for FE and SVM for classification	90.64%
This Research (Phase B)	RF for classification with feature elimination and hyperparameter optimization	94.55%
This Research (Phase A)	Combination of AE, RFE, BO	96.62%

Table 5.1 also provides a comparative relationship of previous work with this proposed model of Phase A. CTG dataset [19] was employed in this research study (for both Phase A and Phase B), was used in all of the research articles shown in the table. This was done to provide a better comparison as well as to highlight the worth of this research by relating it to reputable earlier relevant publications. The unique aspect of this Phase A study is that it suggested a novel model to categorize and predict both fetal condition and morphological patterns using SMOTE, FE, FS, and BO. Despite the fact that earlier research studies have classified and predicted fetal status using a variety of machine learning methods. The novelty of this research lies in the use of many different ML algorithms to complete the objective and in addressing the imbalance between the classes in the CTG dataset [19] while using same dataset (along with the suggest phase A model) for categorizing and estimating status of fetus along with morphological pattern of CTG. According to the findings of this study, the proposed model in Phase A may have therapeutic utility as a tool that can be used to provide support to manage pregnancy related scenarios. This suggested model can assist medical practitioners in making educated judgments and delivering suitable treatment measures, when necessary, by properly detecting and categorizing fetal status and CTG morphological patterns. This practical applicability suggests that suggested phase A mechanism might be implemented in current systems relating to healthcare (that are either

inspired by Sisporto or are direct versions of Sisporto) to help in birth administration, which in turn could improve results relating to pregnancies. Research results supported the hypothesis (in introduction section) for this Phase A. Another advantage of integrating this model into healthcare systems is that it can lessen the negative fetal outcomes. Findings suggest that thorough diagnosis and categorization of prenatal disorders are necessary, especially for highlighting the suspect case and the pathological case with good margin. Moreover, high-risk pregnancies might be managed effectively and in a timely manner while using this model in healthcare systems. It may be possible to identify, diagnose, and promptly resolve difficulties or unfavorable outcomes (for the mother and the fetus) by giving medical personnel a supplementary method to observe high risk pregnancies more efficiently.

For Phase B in this research study, with all 21 features selected and no hyperparameter optimization in part 1, all the classifiers were predicting Normal cases with good confidence levels. Suspicious cases as well as pathologic cases, however, weren't accurately estimated with a high degree of confidence. Particularly, many Suspect instances were being misdiagnosed as Pathologic or Normal. When equated to a category that has normal entries (being mistakenly diagnosed as a suspicious case or a pathologic case), this inaccurate prediction of the Suspect case is more harmful from a medical standpoint. The proper categorization of pathologic cases as well as suspicious cases is more important for CTG categorization than the normal case classification. By feature elimination and hyperparameter optimization, this study achieved good results in fetal status classification. Overall, all classifiers performed better with the proposed mechanism, as observed in Figure 5.2.

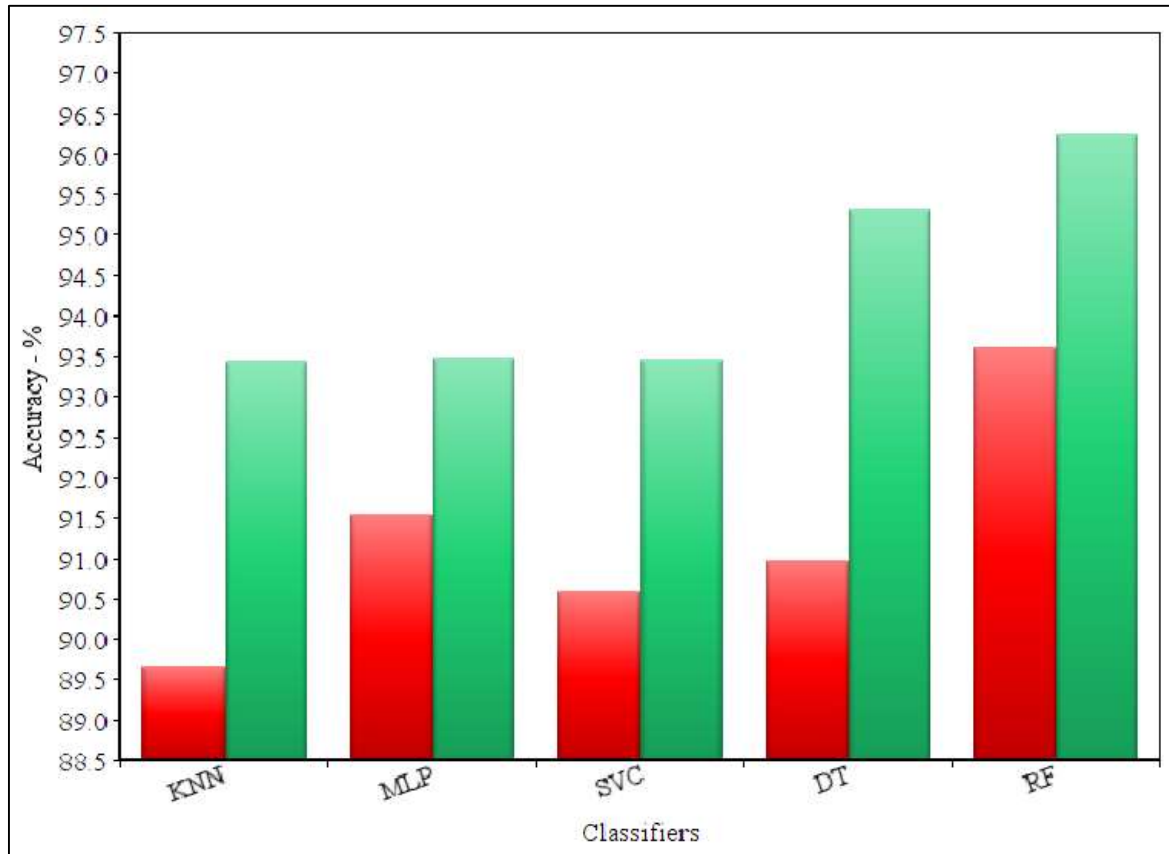


Figure 5.2. Performance Comparison of Classifiers in Phase Part 1 (21 features) in Red and Part 2 (17 features) in Green

All the classifiers performed better for the Normal cases, while using the proposed mechanism. Furthermore, Pathologic cases were comparatively easier to predict and the percentage of incorrect prediction of Suspect cases as Normal was reduced by 60.22% (from 26.4% to 10.5%). The best classifier using the proposed mechanism, to efficiently classify the CTG data and accurately predict the fetal status was Random Forest with an accuracy of 97.20%. From ROC and AUC in Figure 4.11 and Figure 4.12, it can be observed that the proposed mechanism managed to increase the effectiveness of anticipating the pathological cases even better than without using the proposed mechanism. The performance metrics in Table 4.10, Table 4.12, and Table 4.14 show that the suggested mechanism managed to increase the metrics of precision (0.83), recall (0.69), and F1 score (0.76) of the suspect cases to 0.94 (precision), 0.89 (recall), and 0.92 (f1score), that is an increase of 13.25%, 28.98%, and 21.05% respectively. Moreover, pathological and normal cases are predicted with good margins. The significance of this work is that it has achieved the highest accuracy based on the same CTG dataset [19] compared with previous works (Table 5.1). Compared with prior studies in this field, the model accuracy of the proposed model is high due to the fact that

feature elimination and hyperparameter optimization were used together to find the optimal solution. For fair comparison the training to test ratio for this research (phase B) is set to 75:25 for the comparison in Table 5.1.

As evident from Figure 4.9 and Figure 4.10, the most important features in this dataset [19], are ASTV, MSTV, ALTV and Mean. The medical perspective [12] also supports the importance of these features, with emphasis on variability. Static fetal heart rate baselines, loss of fetal heart rate variability, along with the absence of accelerations are signs of fetal hypoxemia and/or acidemia, and these CTG tracing patterns are unsettling as they mean a non-reassuring case [13]. If more precautions are made during CTG monitoring on the elements that deal with these features, then prediction accuracy can be further improved. As observed from Figure 5.3, when using suggested phase B mechanism, overall accuracy of the model was good. Moreover, variation in accuracy of model using different training to testing ratios was not that extreme. However, as observed from Figure 5.4, the importance of using MCC is further emphasized for an imbalanced dataset. Even if accuracy of a model is good, the fact that MCC values during the simulations had an average value of 0.799 and a median value of 0.830 shows that MCC should be considered together with accuracy while dealing with this CTG dataset [19]. Furthermore, the average accuracy and median accuracy across the iterations were 92.65% and 93.65% respectively. Both of the above-mentioned figures show that feature elimination after 4 leads to undesired results.

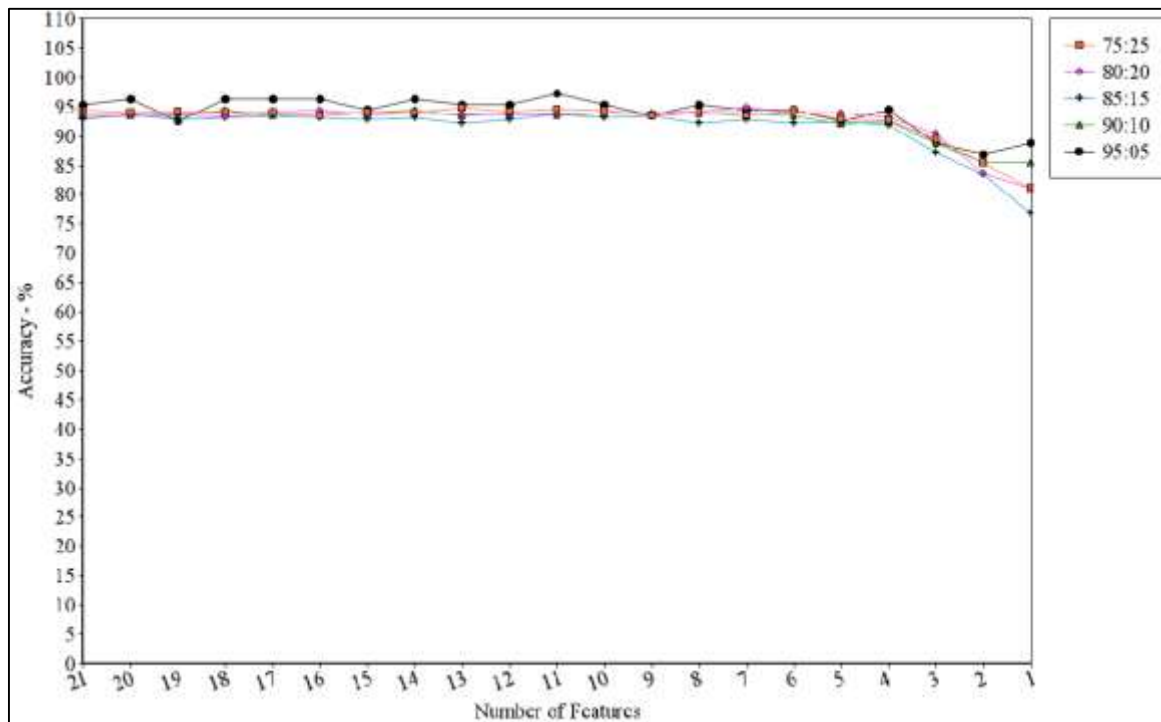


Figure 5.3. Relation of Number of Features with Accuracy

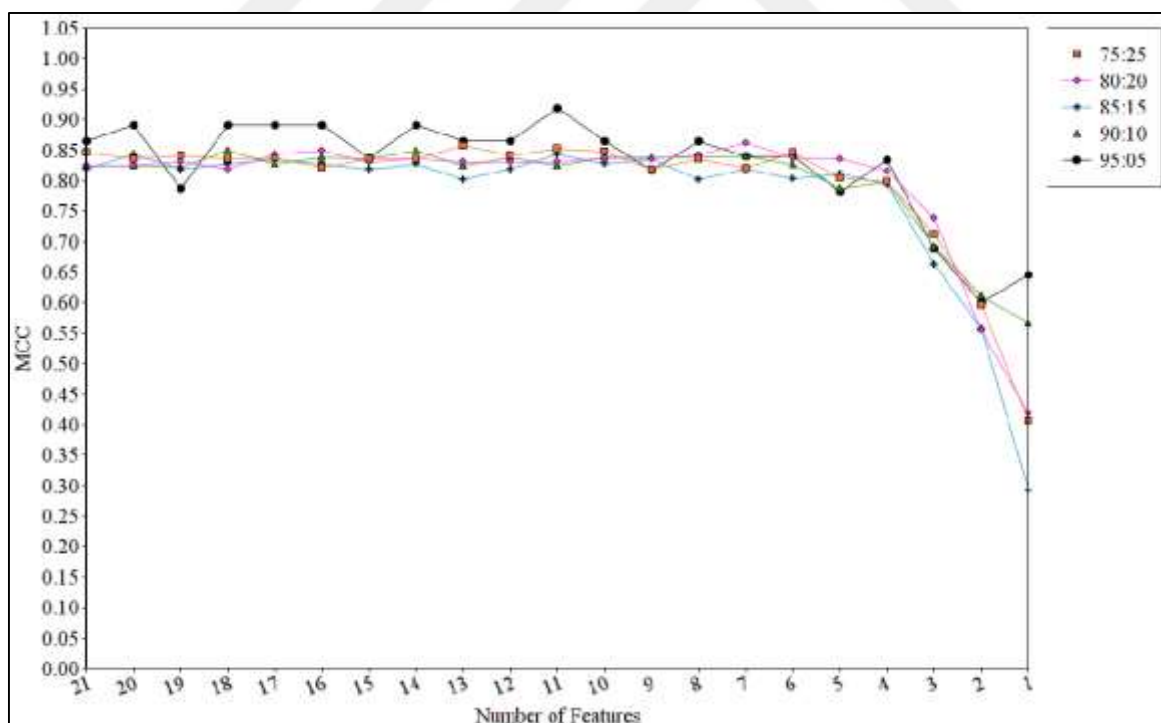


Figure 5.4. Relation of Number of Features with MCC

The final result of the proposed mechanism suggests that the best combination of features (after feature elimination) and optimum hyperparameters for the selected best classifier in this research are as follows:

- Best Selected Classifier in Phase B: Random Forest
- Eliminated number of features: 11.
- Hyperparameters that were altered:
 - The criterion used while splitting was entropy.
 - The tree count in the forest, also known as the estimator count, was 11.
 - For a split at a node, the maximum number of features to check was 9.
 - The weightages are altered so that they are related to the class participation in the input. This mode is also called “Balance” mode for the class weights parameter.
- During the model training, the training - testing ratio was set as 95:05.
- The best MCC value obtained was 0.919.
- The best accuracy obtained was 97.20%.

According to [61], a Maternal COVID-19 infection apparently causes some changes in the CTG trace. These changes can be attributed to multiple causes, the most significant one is the maternal hypoxia. Thus, a stay-at-home mechanism can encourage the patient to avoid coming to the clinic for regular checkups.

Future objectives could also combine more classifiers to further increase the accuracy of the suggested mechanism. The distinctive characteristic of this research of Phase B is that it provides a survey of different relevant popular classifiers for the CTG dataset and then also proposes a new model that utilizes feature elimination along with hyperparameter optimization on the best classifier selected.

To judge the quality of the proposed model in Phase B, it was tested on another related CTG dataset (Czech Technical University (CTU) – University Hospital Brno (UHB) [62]. CTU-UHB dataset is a collection of CTG recordings acquired by ultrasound doppler probe, direct scalp measurement or mixture of both. CTU-UHB has raw signals (containing FHR and UC) as well as descriptive features (in the header files of the signals). Both raw signals and descriptive features were used to test the quality of the proposed model of Phase B. The selection criteria to assess the labor outcome (fetal status) depends on the pH feature in this dataset. This dataset can be dealt with a binary classification case if a threshold is used to bifurcate the pH column in the dataset. The threshold causes the entries of pH column to be "abnormal" if $pH < 7.20$ and "normal" if $pH \geq 7.20$. This creates Normal (375 entries) and Abnormal (177 entries) classes.

The descriptive features of the CTU-UHB dataset are:

- BDecf (it is the base deficit in the extracellular fluid),
- pCO₂ (it is the partial pressure of Carbon Dioxide),
- BE (it is the base excess),
- APGAR1 (it is the Appearance, Pulse, Grimace, Activity and Respiration metric with the first 1 min of breathing),
- APGAR5 (it is the Appearance, Pulse, Grimace, Activity and Respiration metric with the first 5 mins of breathing),
- Gest. weeks (it is the gestational age),
- Weight (it is the neonatal weight in gram),
- Sex (it is the sex of the neonatal),
- Age (it is the maternal age),
- Gravidity (it is the number of times pregnancy is achieved for the related case),
- Parity (it is the number of times a mother gives birth to a love neonate),
- Diabetes (it is a disease related to mother),
- Hypertension (it is a disease related to mother),
- Preeclampsia (it is a disease related to mother),
- Meconium (it is to show whether Meconium affected fluid is present or not),
- Presentation (it is to show whether it's a breech or occipital type),
- Induced (it is to show whether it's a stimulated or natural delivery),
- I.stage (it controls the signal length),
- II.stage (it controls the signal length of the second stage),
- Deliv. type (it is to show whether it's a Cesarean section or vaginal delivery type).

In addition to the above-mentioned descriptive features, other features were made from the raw signals containing FHR and UC. To translate the information in the raw time-based signals into descriptive features, the raw signals were used in making additional features like Peak value, Mean, Median, Standard Deviation, Root mean square (RMS), and Peak to RMS value for both FHR as well as UC.

After implementing the proposed model of Phase B on the CTU-UHB dataset [62] only the following 9 features (out of 32 total features) were selected: BDecf, pCO₂, BE, Apgar1,

Apgar5, I.stage, Mean of UC, RMS of UC, Peak to RMS of UC. The SHAP analysis in Figure 5.5 also supports the reason behind this feature elimination as mostly those features were selected that showed a significant impact of the model output.

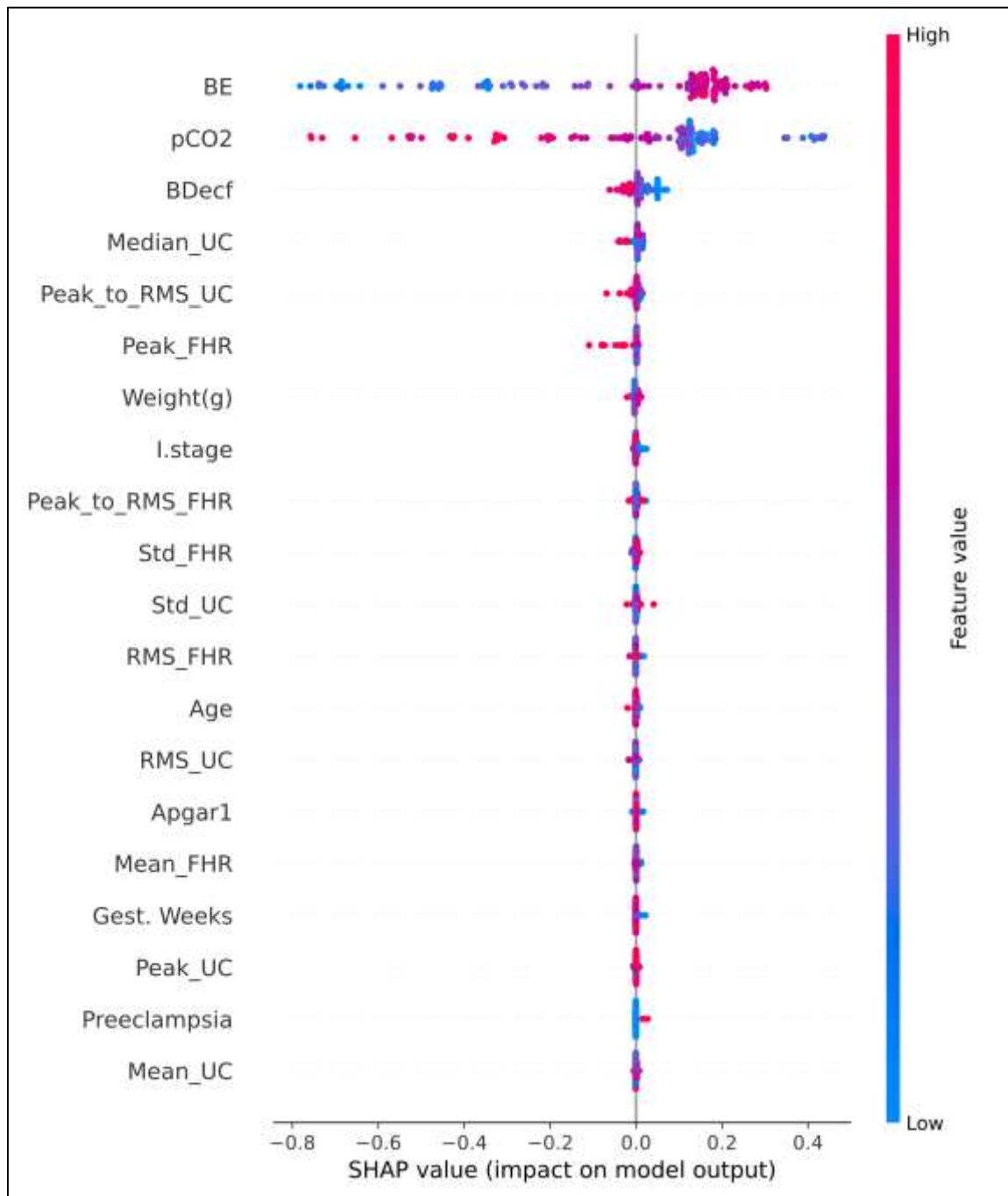


Figure 5.5. SHAP Analysis of CTU-UHB CTG Dataset

The final result of the proposed model of Phase B on the CTU-UHB [62] is as follows:

- Best Selected Classifier in Phase B for CTU-UHB dataset [62]: Random Forest
- Eliminated number of features: 23.
- Hyperparameters that were altered:
 - The criterion used while splitting was entropy.
 - The tree count in the forest, also known as the estimator count, was 50.
 - For a split at a node, the maximum number of features to check was 9.
 - The weightages are altered so that they are related to the class participation in the input. This mode is also called “Balance” mode for the class weights parameter.
- During the model training, the training - testing ratio was set as 85:15.
- The best MCC value obtained was 0.915.
- The best accuracy obtained was 97.34%.

For Phase C, the proposed model performed in predicting the input FECGSYN signal (taken from FECGSYN dataset [20]) as observed from Figure 4.23 (for Case 1), Figure 4.20 (for Case 2) and Figure 4.24 (for Case 4). Moreover, for detecting anomalies (caused by either missing fetal beats or addition of noise or acceleration/deceleration of FHR) in the FECGSYN signals, the proposed model also performed well as observed in Figure 4.17 (for Case 1), Figure 4.21 (for Case 2) and Figure 4.25 (for Case 4). For Case 1 the model loss (as shown in Figure 4.15) performed averagely till the 35th epoch but after the 40th epoch, model was able to learn FECG signals very well. For Case 2 the model loss (as shown in Figure 4.19) performed averagely till the 45th epoch but after the 55th epoch, model was able to learn FECG signals very well. For Case 4 the model loss (as shown in Figure 4.23) performed averagely till the 55th epoch but after the 60th epoch, model was able to learn FECG signals very well. The low MSE and RMSE values depicted (in Table 4.16) show that suggested phase C model was able to estimate the FECG signals quite well. Threshold value was altered by checking the plots of FECG signals. Since Case 1 did not have a huge effect on the FECG baseline thus less anomalies were present. That is the reason that a comparatively larger threshold value was used that was also utilized for all the different types of SNR in Case 1. Threshold value had a relatively smaller value for Case 2 because in this case FHR affected FECG relatively more than when only fetal movements were present in the FECG signals. Whereas comparatively smallest threshold value was used for Case 4. Because

skipped or extra beats in the FECG signals resulted in a considerable rise in the anomaly numbers.

All datasets used in this research work are publicly available (Sisporto CTG dataset [19], CTU-UHB CTG dataset [62], and FECGSYN dataset [20]), so that it is easy for other researchers to compare the results of their research works with the results of this research work. Related relevant work has been published in [63-65].



6. CONCLUSION

Phase A related practical clinical implication is that the suggested methodology may be used to conduct remote CTG telemonitoring of fetal health. Patients would no longer need to attend hospitals and clinics throughout the third trimester. The three fetal states are the major focus of conventional study in this field utilizing dataset of CTG [19]. However, the categorization of morphological patterns of CTG and the relation with fetal states is main emphasis of this research, and subsequently adding to the body of information about CTG. This approach and the positive results of this proposed Phase A model will aid in directing and inspiring scholars to use this proposed mechanism in upcoming related research studies on CTG observation utilizing neural networks. The results of this study would increase the knowledge base of the subject relating to tracking and observing fetal movements. Suggest phase A mechanism of the research includes parts like dataset balancing, feature extraction, feature selection, and Bayesian optimization. Important distinction between this research and the previous relevant research work is that in this research morphological patterns of CTG as well as fetal condition are considered at the same time and also using all the modules discussed above.

Phase B presented a mechanism to classify and predict the different statuses of fetus using CTG. In Part 1, a control experiment was performed to observe the performance of all the selected classifiers using all 21 features. In Part 2, static feature elimination was implemented on all the classifiers. In Part 3, the proposed mechanism (along with a dynamic feature elimination) was used with the best classifier of Part 2 (RF) and was further fine-tuned, via hyperparameter optimization. The clinical application of the current study is that online long distance based fetal health monitoring may be accomplished utilizing the suggested technique in combination with Random Forest classifier. In the third trimester, this would lessen the need for women to frequent clinics and hospitals, which is advantageous in any epidemic (or even pandemic for that matter). Moreover, this proposed method can be used as a decision support system to doctors to lessen their workload. This proposed method and its results will aid in directing and inspiring academics to use the adjusted hyperparameters of this suggested mechanism in their future research studies on neural network related CTG observation systems. The net productivity of subsequent effort in this subject would increase as a result of this knowledge.

Phase C related results supplement the research statement suggested in the introduction section. The simulated FECGSYN [20] signals can be estimated by LSTM based models. Moreover, LSTM can also perform anomaly detection on those signals. Main objective of phase C of this study was to implement a dynamic mechanism that could present a good study of FECG signals. For the value of threshold, an automated system could be implemented for future work. Moreover, this study could be updated to identify the whole QRS along with the anomaly type related to that particular QRS. Furthermore, multiple LSTM types (like Bidirectional LSTM as well as Attention-based LSTM) could be utilized on this proposed phase C model. Standard LSTM provides a sufficient equality between above mentioned LSTM types. Contextual data can be observed easily by bidirectional LSTM, but this causes more time and resource usage. However, specific parts of a data are easily observed by attention-based LSTM, but this can cause the model to sometimes ignore the whole data. Although this approach is good for targeting QRS complex based anomaly detection in the case of FECG.

REFERENCES

1. De-Campos, D. A., Spong, C.Y., and Chandrachan, E. (2015). FIGO consensus guidelines on intrapartum fetal monitoring: Cardiotocography. *International Journal of Gynecology and Obstetrics*, 131, 13–24. <https://doi.org/10.1016/j.ijgo.2015.06.020>.
2. Grivell, R.M., Alfirevic, Z., Gyte, G.M., and Devane, D. (2015). Antenatal cardiotocography for fetal assessment. *The Cochrane database of systematic reviews*. <https://doi.org/10.1002/14651858.CD007863.pub4>.
3. Schneider, U., Schleussner, E., Fiedler, A., Jaekel, S., Liehr, M., Haueisen, J., and Hoyer, D. (2009). Fetal heart rate variability reveals differential dynamics in the intrauterine development of the sympathetic and parasympathetic branches of the autonomic nervous system. *Physiological Measurement*, 215–226. <https://doi.org/10.1088/0967-3334/30/2/008>.
4. Jepsen, I., Blix, E., Cooke, H., Adrian, S.W., and Maude, R. (2022). The overuse of intrapartum cardiotocography (CTG) for low-risk women: An actor-network theory analysis of data from focus groups. *Women Birth*, 35, 593–601. <https://doi.org/10.1016/j.wombi.2022.01.003>.
5. Nazir, L., Lakhta, G., Anees, K., Khan, F.R., Safdar, S., Nazir, G.R., Irum, M.I., Khattak, S.U., and Salim, A. (2021). Admission Cardiotocography as a predictor of low Apgar score: An observational, cross-sectional study. *Cureus*, 13, e14530. <https://doi.org/10.7759/cureus.14530>.
6. Nadeem, G., Rehman, A., and Bashir, H. (2021). Risk Factors Associated With Birth Asphyxia in Term Newborns at a Tertiary Care Hospital of Multan, Pakistan. *Cureus*, 13, e18759. <https://doi.org/10.7759/cureus.18759>.
7. Internet: Perinatal Asphyxia. (2022). WHO. URL: <https://www.who.int/teams/maternal-newborn-child-adolescent-healthand-ageing/newborn-health/perinatal-asphyxia>, Last accessed on 13.05.2023.
8. Internet: Monitoring Health for the SDGs, Sustainable Development Goals. (2022). WHO *World Health Statistics*. URL: <https://www.who.int/publications/i/item/9789240051157>, Last accessed on 13.05.2023.
9. Gregory, E.C.W., Valenzuela, C.P., and Hoyert, D.L. (2022). *Fetal Mortality: United States, 2020*. 71(4). <https://dx.doi.org/10.15620/cdc:118420>.
10. Internet: Enforcement Policy for Non-Invasive Fetal and Maternal Monitoring Devices Used to Support Patient Monitoring During the Coronavirus Disease Public Health Emergency. (2020). FDA. URL: <https://www.fda.gov/regulatory-information/search-fda-guidance-documents/enforcement-policy-non-invasive-fetal-and-maternal-monitoring-devices-used-support>, Last accessed on 14.12.2020.

11. O'Sullivan, M.E., Considine, E.C., O'Riordan, M., Marnane, W.P., Rennie, J.M., and Boylan, G.B. (2021). Challenges of developing robust ai for intrapartum fetal heart rate monitoring. *Frontiers in Artificial Intelligence*. <https://doi.org/10.3389/frai.2021.765210>.
12. Eleftheriades, M., Pervanidou, P., and Chrousos, G. (2007). *Encyclopedia of Stress* (2nd Edition). New York: Academic Press, 46–51. <https://doi.org/10.1016/B978-012373947-6.00492-X>.
13. K H Yu, C., and Bower, S. *Twining's Textbook of Fetal Abnormalities* (3rd Edition). New York; Churchill Livingstone, 211–222. <https://doi.org/10.1016/B978-0-7020-4591-2.00010-3>.
14. Ma'sum, M.A., Riskyana, P. D. I, Jatmiko, W., Krisnadhi, A.A., Setiawan, N.A., and Suarjaya, I.M.A.D. (2019). *Improving Deep Learning Classifier for Fetus Hypoxia Detection in Cardiotocography Signal*. International Workshop on Big Data and Information Security (IWBIS), Bali, 51–56. <https://doi.org/10.1109/IWBIS.2019.8935835>.
15. Alsaggaf, W., Cömert, Z., Nour, M., Polat, K., Brdese, H., and Togaçar, M. (2020). Predicting fetal hypoxia using common spatial pattern and machine learning from cardiotocography signals. *Applied Acoustics*, 167, 107429. <https://doi.org/10.1016/j.apacoust.2020.107429>.
16. Manikandan, M., and Vijayakumar, P. (2021). Improving the Performance of Classifiers by Ensemble Techniques for the Premature Finding of Unusual Birth Outcomes from Cardiotocography. *IETE Journal of Research*, 69, 1734–1744. <https://doi.org/10.1080/03772063.2021.1910579>.
17. Garcia-Canadilla, P., Sanchez-Martinez, S., Crispi, F., and Bijmens, B. (2021). Machine Learning in Fetal Cardiology: What to Expect. *Fetal Diagnosis and Therapy*, 47, 363–372. <https://doi.org/10.1159/000505021>.
18. Oikonomou, V.P., Spilka, J., Stylios, C., and Lhostka, L. (2013). *An adaptive method for the recovery of missing samples from FHR time series*. In Proceedings of the 26th IEEE International Symposium on Computer-Based Medical Systems, Porto, Portugal, pp. 337–342. <https://doi.org/10.1109/CBMS.2013.6627812>.
19. Internet: Cardiotocography Dataset. (2010). *UCI Machine Learning Repository*. URL: <https://archive.ics.uci.edu/ml/datasets/cardiotocography>, Last accessed on 23.07.2022.
20. Andreotti F., Behar J., Zaunseder S., Oster J. and Clifford G D. (2016). An Open-Source Framework for Stress-Testing Non-Invasive Foetal ECG Extraction Algorithms. *Physiological Measurement* 5, 627-648. <https://doi.org/10.13026/C21P4T>.
21. Altini, M., Mullan, P., Rooijakkers, M., Gradl, S., Penders, J., Geusens, N., Grieten, L., and Eskofier, B. (2016). *Detection of Fetal Kicks Using Body-Worn Accelerometers During Pregnancy: Trade-offs Between Sensors Number and Positioning*. Conference Proceedings of IEEE Engineering in Medicine and Biology Society. <https://doi.org/10.1109/EMBC.2016.7591928>.

22. Chamidah, N., and Wasito, I. (2015). *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 (ICACSIS). <https://doi.org/10.1109/ICACSIS.2015.7415166>.
23. Campos, D.A., Bernades, J., Garrido, A., Sa, J.M., and Leite, L.P. (2000). SisPorto 2.0: A Program for Automated Analysis of Cardiotocograms. *The Journal of Maternal-Fetal Medicine*, 9, 311–318. [https://doi.org/10.1002/1520-6661\(200009/10\)9:5<311::AID-MFM12>3.0.CO;2-9](https://doi.org/10.1002/1520-6661(200009/10)9:5<311::AID-MFM12>3.0.CO;2-9).
24. Rahmayanti, N., Pradani, H., Pahlawan, M., and Vinarti, R. (2022). Comparison of machine learning algorithms to classify fetal health using cardiotocogram data. *Procedia Computer Science*, 197, 162–171. <https://doi.org/10.1016/j.procs.2021.12.130>.
25. Noor, N.F.M., Ahmad, N., and Noor, N.M. (2021). *Fetal Health Classification Using Supervised Learning Approach*. IEEE National Biomedical Engineering Conference (NBEC), Virtual, 9–10, 36–41. <https://doi.org/10.1109/NBEC53282.2021.9618748>.
26. Batra, A., Chandra, A., and Matoria, V. (2017). *Cardiotocography Analysis Using Conjunction of Machine Learning Algorithms*. International Conference on Machine Vision and Information Technology, Singapore. <https://doi.org/10.1109/CMVIT.2017.27>.
27. Hoodbhoy, Z., Noman, M., Shafique, A., Nasim, A., Chowdhury, D., and Hasan, B. (2019). Use of Machine Learning Algorithms for Prediction of Fetal Risk using Cardiotocographic Data. *International Journal of Applied and Basic Medical Research*, 9, 226–230. https://doi.org/10.4103/ijabmr.ijabmr_370_18.
28. Agrawal, K., and Mohan, H. (2019). *Cardiotocography Analysis for Fetal State Classification Using Machine Learning Algorithms*. International Conference on Computer Communication and Informatics (ICCCI), Coimbatore, 1–6. <https://doi.org/10.1109/ICCCI.2019.8822218>.
29. Pradhan, A.K., Rout, J.K., Maharana, A.B., Balabantaray, B.K., and Ray, N.K. (2021). *A Machine Learning Approach for the Prediction of Fetal Health using CTG*. 19th OITS International Conference on Information Technology (OCIT), Bhubaneswar, 239–244. <https://doi.org/10.1109/OCIT53463.2021.00056>.
30. Ramla, M., Sangeetha, S., and Nickolas, S. (2018). *Fetal Health State Monitoring Using Decision Tree Classifier from Cardiotocography Measurements*. Second International Conference on Intelligent Computing and Control Systems (ICICCS), Madurai, 1799–1803. <https://doi.org/10.1109/ICCONS.2018.8663047>.
31. Cömert, Z., and Kocamaz, A.F. (2017). A study of artificial neural network training algorithms for classification of cardiotocography signals. *Bitlis Eren University Journal of Science and Technology* 7, 93–103. <https://doi.org/10.17678/beuscitech.338085>.
32. Reddy, G.T., Reddy, M.P.K., Lakshmana, K., Kaluri, R., Rajput, D.S., Srivastava, G., and Baker, T. (2020). Analysis of Dimensionality Reduction Techniques on Big Data. *IEEE Access*, 8, 54776–54788. <https://doi.org/10.1109/ACCESS.2020.2980942>.

33. Kadhim, N.J.A., and Abed, J.K. (2020). *Enhancing the Prediction Accuracy for Cardiotocography (CTG) using Firefly Algorithm and Naïve Bayesian Classifier*. IOP Conference Series: Materials Science and Engineering, 745, 012101. <https://doi.org/10.1088/1757-899x/745/1/012101>.
34. Feng, J., Liang, J., Qiang, Z., Li, X., Chen, Q., Liu, G., Hong, J., Hao, Z., and Wei, H. (2021). *Effective techniques for intelligent cardiotocography interpretation using XGB-RF feature selection and stacking fusion*. IEEE International Conference on Bioinformatics and Biomedicine (BIBM), Houston, 2667–2673. <https://doi.org/10.1109/BIBM52615.2021.9669694>.
35. Dwivedi, P., Khan, A.A., Mugde, S., and Sharma, G. (2021). *Diagnosing the major contributing factors in the classification of the fetal health status using cardiotocography measurements: An AutoML and XAI approach*. 13th International Conference on Electronics, Computers and Artificial Intelligence (ECAI), Pitesti, 1–6. <https://doi.org/10.1109/ECAI52376.2021.9515033>.
36. Piri, J., Mohapatra, P., and Dey, R. (2021). *Multi-objective Ant Lion Optimization Based Feature Retrieval Methodology for Investigation of Fetal Wellbeing*. Third International Conference on Inventive Research in Computing Applications (ICIRCA), Coimbatore, 1732–1737. <https://doi.org/10.1109/ICIRCA51532.2021.9544860>.
37. Lal, A.K., and Karthikeyan, S. (2022). *Deep Learning Classification of Fetal Cardiotocography Data with Differential Privacy*. International Conference on Connected Systems and Intelligence (CSI), Trivandrum, 1–5. <https://doi.org/10.1109/CSI54720.2022.9924087>.
38. Chen, M., and Yin, Z. (2022). *Classification of Cardiotocography Based on the Apriori Algorithm and Multi-Model Ensemble Classifier*. *Frontiers in Cell and Developmental Biology*, 10, 888859. <https://doi.org/10.3389/fcell.2022.888859>.
39. Zhang, Y., and Zhao, Z. (2017). *Fetal State Assessment Based on Cardiotocography Parameters Using PCA and AdaBoost*. 10th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics, Shanghai. <https://doi.org/10.1109/CISP-BMEI.2017.8302314>.
40. Fei, Y., Huang, X., Chen, Q., Chen, J., Li, L., Hong, J., Hao, Z., and Wei, H. (2020). *Automatic Classification of Antepartum Cardiotocography Using Fuzzy Clustering and Adaptive Neuro -Fuzzy Inference System*. IEEE International Conference on Bioinformatics and Biomedicine (BIBM), Seoul, 1938–1942. <https://doi.org/10.1109/BIBM49941.2020.9313143>.
41. Huddar, P.P., and Sontakke, S.A. (2019). *Acquiring Domain Knowledge for Cardiotocography: A Deep Learning Approach*. 3rd International Conference on Informatics and Computational Sciences (ICICoS), Semarang, 1–6. <https://doi.org/10.1109/ICICoS48119.2019.8982397>.

42. Antony, X.R. (2015). *Classification of Fetal Cardiotocography*. Published Master's Thesis, Central Connecticut State University, New Britain.
43. Stylios, I.C., Vlachos, V., and Androulidakis, I. (2014). *Performance comparison of machine learning algorithms for diagnosis of cardiotocograms with class inequality*. 22nd Telecommunications Forum (TELFOR), Belgrade, 951–954. <https://doi.org/10.1109/TELFOR.2014.7034563>.
44. Cömert, Z., Kocamaz, A.F., and Güngör, S. (2016). *Cardiotocography Signals with Artificial Neural Network and Extreme Learning Machine*. 24th Signal Processing and Communication Application Conference (SIU), Zonguldak. <https://doi.org/10.1109/SIU.2016.7496034>.
45. Mertes, G., Long, Y., Liu, Z., Li, Y., Yang, Y., and Clifton, D.A. (2022). A Deep Learning Approach for the Assessment of Signal Quality of Non-Invasive Foetal Electrocardiography. *Sensors*, 22, 3303. <https://doi.org/10.3390/s22093303>.
46. Fotiadou, E., Xu, M., van Erp, B., van Sloun, R. J. G., and Vullings, R. (2020). *Deep Convolutional Long Short-Term Memory Network for Fetal Heart Rate Extraction*. Annual International Conference of the IEEE Engineering in Medicine and Biology Society, 608–611, Canada. <https://doi.org/10.1109/EMBC44109.2020.9175442>.
47. Zhou, Z., Huang, K., Shen, H., Qiu, Y., and Z. Ming. (2021). Morphology Extraction of Fetal Electrocardiogram by Slow-Fast LSTM Network. *Biomedical Signal Processing and Control*, 68, 102664. <https://doi.org/10.1016/j.bspc.2021.102664>.
48. Wilson, J. D., Govindan, R. B., Hatton, J. O., Lowery, C. L., and Preissl, H. (2008). Integrated Approach for Fetal QRS Detection. *IEEE Transactions on Biomedical Engineering*, 55, 9, 2190-2197. <https://doi.org/10.1109/TBME.2008.923916>.
49. Wang, B. and Saniie, J. (2018). *Fetal Electrocardiogram Recognition Using Multilayer Perceptron Neural Network*. IEEE International Conference on Electro/Information Technology (EIT), Rochester, 0434-0437. <https://doi.org/10.1109/EIT.2018.8500232>.
50. Bandurova, S., Biloborodova, T., Skarha-Bandurov, I., Boltov, Y., and Derkach, M. (2021). A Multilayer LSTM Auto-Encoder for Fetal ECG Anomaly Detection. *Studies in Health Technology and Informatics*, 285, 147-152. <https://doi.org/10.3233/SHTI210588>.
51. Internet: Tong, Y., and Hong, Z. (2020). Hyper-Parameter Optimization: A Review of Algorithms and Applications. *Cornell University arXiv*. URL: <http://arXiv:cs.LG/2003.05689>, Last accessed on 15.04.2023.
52. Johansen, A. (2010). *International Encyclopedia of Education* (3rd Edition). London: Elsevier, 296–303. <https://doi.org/10.1016/B978-0-08-044894-7.01543-8>.

53. Mockus, J. (1974). *On Bayesian methods for seeking the extremum*. Optimization Techniques IFIP Technical Conference, Novosibirsk, 400–404. https://doi.org/10.1007/3-540-07165-2_55.
54. Chawla, N.V., Bowyer, K.W., Hall, L.O., and Kegelmeyer, W.P. (2002). SMOTE: Synthetic Minority Over-sampling Technique. *Journal of Artificial Intelligence Research*, 16, 321–357. <https://doi.org/10.1613/jair.953>.
55. Yin, Y., and Bingi, Y. (2023). Using Machine Learning to Classify Human Fetal Health and Analyze Feature Importance. *BioMed Informatics*, 3, 280–298. <https://doi.org/10.3390/biomedinformatics3020019>.
56. Elm, E. V., Altman, D.G., Egger, M., Pocock, S.J., Gøtzsche, P.C., and Vandenbroucke, J.P. (2007). The Strengthening the Reporting of Observational Studies in Epidemiology (STROBE) statement: guidelines for reporting observational studies. *Annals of Internal Medicine*, 147, 573–577. <https://doi.org/10.7326/0003-4819-147-8-200710160-00010>.
57. Das, S., Mukherjee, H., Roy, K., and Saha, C.K. (2023). Fetal Health Classification from Cardiotocograph for Both Stages of Labor A Soft-Computing-Based Approach. *Diagnostics*, 13, 858. <https://doi.org/10.3390/diagnostics13050858>.
58. De-Campos, D.A., Rei, M., Nunes, I., Sousa, P., and Bernardes, J. (2017). SisPorto 4.0—computer analysis following the 2015 FIGO Guidelines for intrapartum fetal monitoring. *The Journal of Maternal-Fetal and Neonatal Medicine*, 30, 62–67. <https://doi.org/10.3109/14767058.2016.1161750>.
59. Bernardes, J. (2023). Computerized analysis of cardiotocograms in clinical practice and the SisPorto® system thirty-two years after: Technological, physiopathological and clinical studies. *Journal of Perinatal Medicine*, 51, 145–160. <https://doi.org/10.1515/jpm-2022-0406>.
60. Lopes-Pereira, J., Costa, A., De-Campos, D. A., Costa-Santos, C., Amaral, J., and Bernardes, J. (2019). Computerized analysis of cardiotocograms and ST signals is associated with significant reductions in hypoxic-ischemic encephalopathy and cesarean delivery: an observational study in 38,466 deliveries. *American Journal of Obstetrics and Gynecology*, 220, 269.e1–269.e8. <https://doi.org/10.1016/j.ajog.2018.12.037>.
61. Gracia-Perez-Bonfils, A., Martinez-Perez, O., Llubra, E., and Chandrabaran, E. (2020). Fetal heart rate changes on the cardiotocograph trace secondary to maternal covid-19 infection. *European Journal of Obstetrics and Gynecology and Reproductive Biology*, 252, 286–293. <https://doi.org/10.1016/j.ejogrb.2020.06.049>.
62. Chudacek, V., Spilka, J., Bursa, M., Janku, P., Hruban, L., Huptych, M., and Lhotska, L. (2014). Open access intrapartum ctg database. *BMC Pregnancy and Childbirth*, 14(16), 1–12. <https://doi.org/10.13026/C22013>.
63. Akmal, H., Hardalaç, F., and Ayturan, K. (2023). A Fetal Well-Being Diagnostic Method Based on Cardiotocographic Morphological Pattern Utilizing Autoencoder and Recursive Feature Elimination. *Diagnostics*, 13(11), 1931. <http://doi.org/10.3390/diagnostics13111931>.

64. Akmal, H., Ayturan. K., and Hardalaç. F. (2023). *Predicting And Detecting Abnormalities In Fetal Ecg Signals Using Lstm Based Neural Networks*. 9th International Congress on Innovative Scientific Approaches, Samsun.
65. Akmal, H., Ayturan. K., and Hardalaç. F. (2023). *A Fetal Well-Being Decision Support System based on Cardiotocographic Morphological Pattern Utilizing Feature Extraction and Selection*. 9th International Scientific Research Congress of Black Sea Coastal Countries, Ankara.







Gazili olmak ayrıcalıktır