A Project report

on

# CLASSIFICATION ON CARDIOTOCOGRAPHY (CTGs)

BY

**Ashwin Srinath Sureshkumar**

# ABSTRACT

CLASSIFICATION ON CARDIOTOCOGRAPHY (CTGs)

The project aims at classifying the CTGs of medical patients into three main categories: normal (no risk), suspect (possibility of risk) and pathologic (under risk) and predicting the medical state of a patient. This classification is performed using several classifiers including SVM and Neural network classifiers. The performance evaluation of each of the classifiers is done and then a comparison between them is noted.

Cardiotocography (CTG) is a simultaneous recording of fetal heart rate (FHR) and uterine contractions (UC) which help in predicting fetal state. It is one of the most common techniques to evaluate maternal and fetal well-being during pregnancy and before delivery.

The dataset consists of 2126 samples of data with fetal state classes: (1=normal, 2=suspect, 3=pathologic) as labels and 23 features:
(LB - FHR baseline (beats per minute), AC - # of accelerations per second, FM - # of fetal movements per second, UC - # of uterine contractions per second, DL - # of light decelerations per second, DS - # of severe decelerations per second, DP - # of prolonged decelerations per second, ASTV - percentage of time with abnormal short term variability, MSTV - mean value of short term variability, ALTV - percentage of time with abnormal long term variability, MLTV - mean value of long term variability, Width - width of FHR histogram, Min - minimum of FHR histogram, Max - Maximum of FHR histogram, Nmax - # of histogram peaks, Nzeros - # of histogram zeros, Mode - histogram mode, Mean - histogram mean, Median - histogram median, Variance - histogram variance, Tendency - histogram tendency, CLASS - FHR pattern class code (1 to 10), NSP - fetal state class code (N=normal; S=suspect; P=pathologic)
The dataset used is from https://archive.ics.uci.edu/ml/datasets/Cardiotocography under the UCI Machine Learning Repository

The machine learning algorithms intended to be used for the classification are: Neural network, AdaBoost, Gradient Boosting, Perceptron, KNN, Support Vector Machine (SVM), Decision tree and Random forest algorithms. The metrics used to evaluate performance of these algorithms would be: Classification Accuracy, Confusion Matrix, F1 Score, Mean Absolute Error, Mean Squared Error or any other metric specific to an algorithm.

This project provides assistance for the obstetricians in classifying the patient CTGs and assessing the state of the fetus better, thus taking an action appropriately.


**Keywords:** Cardiotocogram, CTG, Machine Learning, Classifiers, fetal state, dataset, metrics.

# TABLE OF CONTENTS

# CHAPTER-I
# INTRODUCTION

During pregnancy, it is not possible to get direct information about fetus. Therefore, obstetricians rely on indirect information about fetal condition. One of the most exploited information is fetal heart rate (FHR). Almost important restriction of electronic fetal heart rate monitoring (EFM) interpretation has been an unacceptably high inter- and intra- observer variation in interpretation. Such variation hinders the important clinical goals of accurate communication and application of timely management. Recent efforts to diminish the problems of delayed interference for abnormal tracings have resulted in a range of methods to categorize tracings and guide management.

Currently, Cardiotocography is the most spread indirect, non-invasive diagnostic technique, in daily clinical practice, to monitor fetal health. FHR and uterine contractions (UC) are simultaneously recorded by means of two probes placed on the maternal abdomen: a US Doppler probe for FHR signal and a pressure transducer for UC signal. Cardiotocogram (CTG) contains of two distinct signals, its continuous recording of instantaneous fetal heart rate (FHR) and uterine activity (UC). Only during labor, after spontaneous or induced membrane rupture, direct measurement of intra-uterine pressure and fetal ECG can provide more accurate results. The information which is acquired from CTG is used for early recognition of a pathological state (i.e. congenital heart defect, fetal distress or hypoxia, etc.) and may help the obstetrician to predict future complications and interpose before there is a permanent damage to the fetus.

Although its usefulness, there has been some disagreement as to the utility and the effectiveness of CTG observing, especially in low-risk pregnancies. Still nowadays, there is a very high intra- and inter-observer fluctuation in the assessment of FHR patterns, which can lead to an incorrect appraisal of fetal status. On one hand a falsely diagnosed fetal pain may lead to unnecessary interventions; on the other hand, an improper diagnosis of fetal well-being may deny necessary maintenances. To advance CTG analysis, more objective methods for CTG interpretation are of vital importance; therefore, significant efforts have been spent and several analysis approaches have been proposed in recent years. The purpose of this study is to present a comparison between two different techniques for ECG recordings. In this study, I used machine learning algorithms and classifiers to classify CTGs and compared their performances with metrics.

# CHAPTER-II
# PROBLEM STATEMENT

## 2.1 Problem description:

The need to classify the patients based on their symptoms and conditions drives this project. The aim is to classify into three fetal states based on patient data. We use multiple Machine learning classifiers and note down the performances of them so that we can conclude as to which classifier is able to perform well for the data set and which classifier is suitable for such classification tasks.

## 2.2 Objective:

The main objective is to understand the implementations of Machine learning classifiers through the problem statement stated and inculcate the knowledge of the classifiers in this scenario. Also, the project aims to elaborate the usage of Machine learning in the field of health informatics and the advances happening in the field.

## 2.3 Proposed system:

The project consists of implementation of Machine learning process in sequence that includes data set loading, data extraction, training and testing the data with classifiers. The classifiers include Perceptron, Decision tree classifier, Support vector classifier, KNN classifier , multi-layer perceptron and ensemble learning methods. After training on these classifiers we predict the fetal states of patients on test data and calculate the performance metrics. After the calculations and predictions we decide as to which classifier is suitable for this data set and scenario. This gives us the understanding of the suitability of a classifier in a given problem statement.

CTG signals -> Data and Feature extraction -> Classifier application -> Prediction

## 2.4 System requirements:

- UC Irvine Cardiotocography dataset
- Python work environment (any Python editor would though Jupyter notebook has been used for the project)
- An operating system compatible to run python program
- Installation of necessary machine learning python packages.

# CHAPTER-III
# DATASET DESCRIPTION

The dataset consists of 2126 samples of data with fetal state classes: (1=normal, 2=suspect, 3=pathologic) as labels and 23 features:

(LB - FHR baseline (beats per minute),
AC - # of accelerations per second,
FM - # of fetal movements per second,
UC - # of uterine contractions per second,
DL - # of light decelerations per second,
DS - # of severe decelerations per second,
DP - # of prolonged decelerations per second,
ASTV - percentage of time with abnormal short term variability,
MSTV - mean value of short term variability,
ALTV - percentage of time with abnormal long term variability,
MLTV - mean value of long term variability,
Width - width of FHR histogram,
Min - minimum of FHR histogram,
Max - Maximum of FHR histogram,
Nmax - # of histogram peaks,
Nzeros - # of histogram zeros,
Mode - histogram mode,
Mean - histogram mean,
Median - histogram median,
Variance - histogram variance,
Tendency - histogram tendency,
CLASS - FHR pattern class code (1 to 10),
NSP - fetal state class code (N=normal; S=suspect; P=pathologic)

The NSP fetal state class code helps us in defining the medical state of the patient chosen.

The dataset used is from https://archive.ics.uci.edu/ml/datasets/Cardiotocography under the UCI Machine Learning Repository

# CHAPTER-IV
# DATA PRE-PROCESSING

## 4.1 Missing values:

Having missing values in a dataset is a common trait to be encountered with. The simplest strategy for handling missing data is to remove records that contain a missing value.We can do this by creating a new Pandas DataFrame with the rows containing missing values removed. Pandas provides the dropna() function that can be used to drop either columns or rows with missing data.It is an easy way for the dataset used in the project as the number of rows with missing values were very few and dropping those rows did not greatly affect the shape of the dataset.

## 4.2 Feature scaling:

Since, most of the machine learning algorithms use Eucledian distance between two data points in their computations, this is a problem. In our project too classifiers like KNN have a significant change in performance due to feature scaling. Performing feature scaling is one of the important steps in the project as it greatly contributed to the improvement of accuracies for classifiers. It also indicates the difference in magnitude ranges of features and how high different magnitude scales are found in the dataset.

## 4.3 Principal Component analysis:

In the project dimensionality reduction has been performed using principal components and reduced the dimensions to 18 maintaining a variance of 95% in the data. Sometimes, most of these features are correlated, and hence redundant. This is where dimensionality reduction algorithms come into play.

## 4.4 Accuracy calculations:

As it is a multi-class dataset, accuracy scores may not be the right way of measuring the classifier performance. We can use multiple methods like down sampling/up sampling or penalizing algorithms/AUROC scores. So we use AUROC scores for this project instead of general accuracy scores. However we cannot easily get roc scores for multi-class problems. Thus we use a Label binarizer for the purpose. A simple way to extend these algorithms to the multi-class classification case is to use the so-called one-vs-all scheme. At learning time, this simply consists in learning one regressor or binary classifier per class. In doing so, one needs to convert multi-class labels to binary labels (belong or does not belong to the class). LabelBinarizer makes this process easy. with the transform method. At prediction time, one assigns the class for which the corresponding model gave the greatest confidence. LabelBinarizer makes this easy with the inverse_transform method.

# CHAPTER-V
# RESULTS

## 5.1 Accuracy values:

| CLASSIFIER | AUROC SCORES |
|---|---|
| | |
| K neighbors | 0.9752 |
| | |
| Decision tree | 0.9682 |
| | |
| Random forest | 0.9813 |
| | |
| Ada boost | 0.9563 |
| | |
| Gradient boosting | 0.9694 |
| | |
| Support vector classifier | 0.9845 |
| | |
| Perceptron | 0.9687 |
| | |
| Multi-layer perceptron | 0.9752 |

## 5.2 Comparison results

Clearly support vector classifier has performed the best of all and Ada boost classifier performed the least. Tree-based models were just fine but not that upto-the mark. Perceptron classifier could not give a great accuracy but performed well than tree based models. Variation in iterations value and alpha value could not affect accuracy greatly. Support vector classifier performed well using rbf kernel and gave a good accuracy. Though it recorded almost highest accuracy of all algorithms, it is greatly affected by feature scaling and could be sensitive to the data used. So this classifier alone cannot be recommended for fetal state classification and any hybrid classifiers working along with SVM would be a better choice.

Needless to say that from the accuracies recorded, decision tree could not perform that well. Despite changing its parameters, it could not improve any further. This is the least recommended classifier of all for CTG classification. AdaBoost and Gradient boost classifiers could not perform well for the data provided. Random forest classifier gave a good accuracy when decision tree is used as base estimator but is still sensitive to the data set used and does not have a huge scope for improvement after a point. Multilayer perceptron gave a very good accuracy of all the classifiers. Working on MLP could produce better results over the time.

# CHAPTER-VI
# CLASSIFIERS

## 6.1 Perceptron:

A perceptron is defined as a machine that can learn, using samples, for assigning input vectors (samples) to different classes, using a linear function of the inputs. An individual node which applies a step function to the net weighted sum of its inputs represents such perceptron. The input pattern is regarded as belongs to one class or the other depending on whether the node output is 1 or 0. From the point of view of applied hardware applications, since the weight values have neither too big nor too small, the weights values get the significant importance. We will thus be using this classifier in our project.

<u>Specifications:</u>
Maximum iterations = 1000
Alpha value used= 0.0001(varying it does not show significant change in accuracy)

<u>Behavior:</u>
Perceptron classifier could not give a great accuracy but performed well than tree based models. Variation in iterations value and alpha value could not affect accuracy greatly.

## 6.2 Support vector classifier:

In machine learning classification, SVM finds an optimal hyperplane that best segregates observations from different classes. A hyperplane is a plane of n-1 dimension that separates the n dimensional feature space of the observations into two spaces. For example, the hyperplane in a two-dimensional feature space is a line, and a surface in a three-dimensional feature space. The optimal hyperplane is picked so that the distance from its nearest points in each space to itself is maximized. And these nearest points are the so-called support vectors. For our project especially rbf and linear svm seems to be more suitable.

<u>Specifications:</u>
Value of C= 1
Kernel used = rbf (gives better accuracy than linear kernel)
Degree of polynomial used =3

<u>Behavior:</u>
Support vector classifier performed well using rbf kernel and gave a good accuracy. Though it recorded almost highest accuracy of all algorithms, it is greatly affected by feature scaling and

could be sensitive to the data used. So this classifier alone cannot be recommended for fetal state classification and any hybrid classifiers working along with SVM would be a better choice.

**6.3 Decision tree:**

A decision tree is a tree-like graph, a sequential diagram illustrating all of the possible decision alternatives and the corresponding outcomes. Starting from the root of a tree, every internal node represents what a decision is made based on; each branch of a node represents how a choice may lead to the next nodes; and finally, each terminal node, the leaf, represents an outcome yielded. We will be using this classifier even as a base classifier with ensemble learning methods in the project.

Specifications:
Best split criterion used = Gini impurity
Minimum samples split=2 (varying this did not affect accuracy after a point)

Behavior:
Needless to say that from the accuracies recorded, decision tree could not perform that well. Despite changing its parameters, it could not improve any further. This is the least recommended classifier of all for CTG classification

**6.4 Ensemble methods:**

Ensemble learning helps improve machine learning results by combining several models. This approach allows the production of better predictive performance compared to a single model. We will be using Random forest, Ada boost and Gradient boost classifiers in this project.

Specifications:
Random forest:
Estimators used=100
Criterion=gini impurity
Min samples split=2

AdaBoost:
Base estimator is Decision tree classifier with maximum depth=2
Estimators used=50
Learning rate =1

Gradient Boosting:
Learning rate=0.1
Estimators used =100

Min samples split=2

Behavior:
AdaBoost and Gradient boost classifiers could not perform well for the data provided. Random forest classifier gave a good accuracy when decision tree is used as base estimator but is still sensitive to the data set used and does not have a huge scope for improvement after a point.

### 6.5 Neural networks:

Neural networks are a set of algorithms, modeled loosely after the human brain, that are designed to recognize patterns. They interpret sensory data through a kind of machine perception, labeling or clustering raw input. In this project multi-layer perceptron has been implemented to classify patient data.

Specifications:
Hidden layer sizes=100
Activation used =rectified linear activation
Alpha value=0.0001
Maximum iterations=200

Behavior:
Multilayer perceptron gave a very good accuracy of all the classifiers. Working on MLP could produce better results over the time and the classifier seems to suit the problem statement well. MLP classifier is highly recommended especially for fetal state classifications.

### 6.6 K-nearest neighbors:

KNN is a non-parametric, lazy learning algorithm. Its purpose is to use a database in which the data points are separated into several classes to predict the classification of a new sample point.
This classifier can also help us in classifying into fetal states. So this classifier is also implemented and its performance is noted.

Specifications:
Neighbors =5
Uniform weights used
Euclidean distance used

Behavior:
Being a distance metric based classifier, it is highly sensitive to magnitude ranges of features. Feature scaling is must for it and can greatly improve the accuracy. Though the accuracy cannot be titled as best but it managed to perform better that tree-based models.

# CHAPTER-VII
# CONCLUSION

## 7.1 Project Conclusions:

The three conclusions that can be drawn from the project are as follows:

- Tree based models do not seem to be that appropriate for the dataset or CTG classification in general.
- Neural networks have better scope of performing well and must be explored more.
- Support vector classifier (rbf kernel) is also quite suitable for CTG classifications.

## 7.2 Future enhancements:

The major limitations of this method are: the sifting process used in EMD(electronic signal equipment) is time consuming and the number of decomposed components varies with respect to the signal resulting in some empty spaces in the feature set. At this stage we can state that the results are quite promising. However, to be of clinical significance the proposed methodology requires extensive validation on a bigger data set.

As a future work, the proposed methodology would be applied to FHR signals of different durations and also good accuracies would be extended for multi-class classification. Another possible extension is the classification of FHR by associating the traces to scores. The effect of different sampling rates on features extracted using EMD and its effect on the SVM kind of classifiers also need further investigation. Neural networks have a very good scope to perform well for health informatics and CTG classification specifically. It has to be further dealt with and more experimenting can be done with them.

# CHAPTER-VIII
# REFERENCES


*[1] Jekova, Bortolan, G. & Christov, I., 2008. Assessment and comparison of different methods for heartbeat classification, ScienceDirect, 30. Medical Engineering & Physics, Issue 30, pp. 248-257.*

*[2] Palomäki, O., Luukkaala, T., Luoto, R. & Tuimala, R., 2006. Intrapartum cardiotocography: the dilemma of interpretational variation.. Journal of Prenatal Medical, pp. 298-302.*

*[3] Rosenblatt, F., 1988 . The perception: a probabilistic model for information storage and organization in the brain. Cambridge, MA, USA : MIT Press .*

*[4] Van Geijn, H., 1996. Developments in CTG analysis. Baillière's Clinical Obstetrics and Gynaecology, June, pp. 185-209.*

*[5] https://www.datacamp.com/community/tutorials/principal-component-analysis-in-python*

*[6]https://stats.stackexchange.com/questions/244507/what-algorithms-need-feature-scaling-beside-from-svm*

*[7] https://elitedatascience.com/imbalanced-classes*

*[8] Introduction to Machine Learning, Third Edition, Ethem Alpaydin, MIT Press, 2014*

*[9] Python Machine Learning: Machine Learning and Deep Learning with Python, scikitlearn, and TensorFlow, Second Edition, Sebastian Raschka and Vahid Mirjalili, Packt Publishing, 2017.*