Fetal AI: Fetal Health Classification System

Final Project Report

Prepared For: Smart-Internz Applied Data Science Guided Project

By: Sakshi Santosh Patil

Affiliation: D Y Patil Agriculture and Technical University, Talsande

Date: 18 June 2025

Abstract

FetalAI is a machine learning-based web application that classifies fetal health into Normal, Suspect, or Pathological categories using cardiotocography (CTG) data from the fetal_health.csv dataset (2126 records, 21 features). An op- timized XGBoost model, integrated with a Flask web interface, achieves a test accuracy of ~90% and a macro F1-score of ~0.85. The system provides a user- friendly platform for healthcare providers and expectant parents, with API sup- port for medical system integration, enhancing prenatal care efficiency.

Contents

1 Introduction 4
1.1 Project Overview 4
1.2 Objectives 4

2 Project Initialization and Planning Phase 4
2.1 Problem Statement 4
2.2 Proposed Solution 4
2.3 Initial Project Planning 5

3 Data Collection and Preprocessing Phase 5
3.1 Data Collection Plan and Raw Data Sources 5
3.2 Data Quality Report 5

4 Model Development Phase 5
4.1 Model Selection Report 5
4.2 Initial Model Training Code, Validation, and Evaluation 5

5 Model Optimization and Tuning Phase 7
5.1 Tuning Documentation 7
5.2 Model Performance 7
5.3 Final Model Selection 7

6 Results 7
6.1 Output Description 7

7 Advantages & Disadvantages 10
7.1 Advantages 10
7.2 Disadvantages 10

8 Conclusion 10

9 Future Scope 10

10 Appendix 10
10.1 Source Code 10
10.2 Project Resources 11

1 Introduction

1.1 Project Overview

FetalAI automates fetal health classification using CTG data, leveraging an XGBoost model deployed via a Flask application. The dataset (`fetal_health.csv`) contains 2126 records with 21 features (e.g., `baseline value`, `accelerations`) and a target variable (`fetal_health`: 1=Normal, 2=Suspect, 3=Pathological). The system includes a web interface (`index.html`, `result.html`) and API, supporting prenatal care decision-making.

1.2 Objectives

- Achieve a macro F1-score > 0.85 for fetal health classification.
- Develop an intuitive web interface for CTG input and prediction display.
- Provide API support for medical system integration.
- Reduce specialist workload through automated CTG analysis.

2 Project Initialization and Planning Phase

2.1 Problem Statement

PS No.	I am (Customer)	I'm trying to	But	Because	Which makes me feel
PS-1	Expectant parent	Monitor my baby's health	I lack medical expertise	CTG data is complex	Anxious about fetal safety
PS-2	Obstetrician	Assess fetal health quickly	Manual CTG analysis is slow	It requires specialized skills	Overwhelmed by workload
PS-3	Midwife	Provide accurate prenatal care	I can't interpret CTG data reliably	Limited access to tools	Insecure about patient outcomes

Table 1: Problem Statements for FetalAI

2.2 Proposed Solution

FetalAI employs an XGBoost model trained on preprocessed CTG data, integrated into a Flask web application. Key features include data preprocessing (`StandardScaler`, stratified sampling), model training, and a web interface for user input and predictions.

User Story	Task	Story Points	Priority
USN-1	Preprocess CTG dataset with scaling and splitting	2	High
USN-2	Train XGBoost model on preprocessed data	3	High
USN-3	Integrate XGBoost model with Flask application	3	High
USN-4	Develop web form for CTG input and prediction display	2	High

Table 2: Product Backlog for FetalAI

2.3 Initial Project Planning

3 Data Collection and Preprocessing Phase

3.1 Data Collection Plan and Raw Data Sources

The dataset was sourced from the Kaggle Repository (https://www.kaggle.com/datasets/andrewmvd/fetal-health-classification), containing 2126 records in CSV format (~47 KB) with 21 CTG features and a target variable.

3.2 Data Quality Report

Dataset	Data Quality Issue	Severity	Resolution Plan
fetal_health	Class Imbalance	High	Stratified sampling, class weights
fetal_health	Potential Outliers	Moderate	Identify and cap using IQR
fetal_health	Missing Values (if any)	Low	Impute with mean or remove rows

Table 3: Data Quality Issues for FetalAI

4 Model Development Phase

4.1 Model Selection Report

Selected Model: XGBoost due to its superior test accuracy (~90%) and macro F1-score (~0.85).

4.2 Initial Model Training Code, Validation, and Evaluation

Section	Description
Data Overview	The dataset contains 2126 records with 21 CTG features and a target variable (fetal$_health : 1 = Normal, 2 = Suspect, 3 = Pathological$).
Scaling	Features scaled using `StandardScaler`, saved as `scaler.pkl`.
Handling Class Imbalance	Stratified 80%-20% train-test split, class weights in XGBoost.
Outlier Detection	Outliers identified using IQR, capped or removed.
Target Adjustment	`fetal_health` adjusted from 1,2,3 to 0,1,2 for XGBoost.

Table 4: Data Preprocessing Steps for FetalAI

Model	Description
Logistic Regression	Linear model with multinomial loss, suitable for balanced datasets. Limited for complex relationships.
Random Forest	Ensemble of decision trees, robust to class imbalance. Less efficient for large datasets.
XGBoost	Gradient boosting model, excels in handling class imbalance and complex feature interactions.

Table 5: Models Evaluated for FetalAI

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from xgboost import XGBClassifier
from sklearn.metrics import classification_report
import joblib

df = pd.read_csv('data/fetal_health.csv')
X = df.drop('fetal_health', axis=1)
y = df['fetal_health']
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)
joblib.dump(scaler, 'scaler.pkl')
y_adjusted = y - 1
X_train, X_test, y_train, y_test = train_test_split(X_scaled,
    y_adjusted, test_size=0.2, random_state=42, stratify=y_adjusted)
model = XGBClassifier(use_label_encoder=False, eval_metric='mlogloss
    ', random_state=42)
model.fit(X_train, y_train)
joblib.dump(model, 'fetalai_model.pkl')
y_pred = model.predict(X_test)
```

```
print(classification_report(y_test, y_pred, target_names=['Normal',
    'Suspect', 'Pathological']))
```

Model	Performance Metrics
XGBoost	Test Accuracy: ~90%, Macro F1-Score: ~0.85

Table 6: XGBoost Performance

5 Model Optimization and Tuning Phase

5.1 Tuning Documentation

```
from sklearn.model_selection import GridSearchCV
param_grid = {
    'n_estimators': [100, 200, 300],
    'max_depth': [3, 6, 9],
    'learning_rate': [0.01, 0.1, 0.3],
    'subsample': [0.7, 0.8, 1.0]
}
grid_search = GridSearchCV(XGBClassifier(use_label_encoder=False,
    eval_metric='mlogloss', random_state=42), param_grid, cv=5,
    scoring='f1_macro', n_jobs=-1)
grid_search.fit(X_train, y_train)
joblib.dump(grid_search.best_estimator_, 'fetalai_model.pkl')
```

Optimized parameters: `n_estimators=200`, `max_depth=6`, `learning_rate=0.1`, `subsample=0.8`.

5.2 Model Performance

Model	Test Accuracy	Macro F1-Score
XGBoost (Baseline)	~88%	~0.82
XGBoost (Optimized)	~90%	~0.85

Table 7: Optimized XGBoost Performance

5.3 Final Model Selection

The optimized XGBoost model was selected for its improved accuracy and F1-score, enhancing performance on minority classes (Suspect, Pathological).

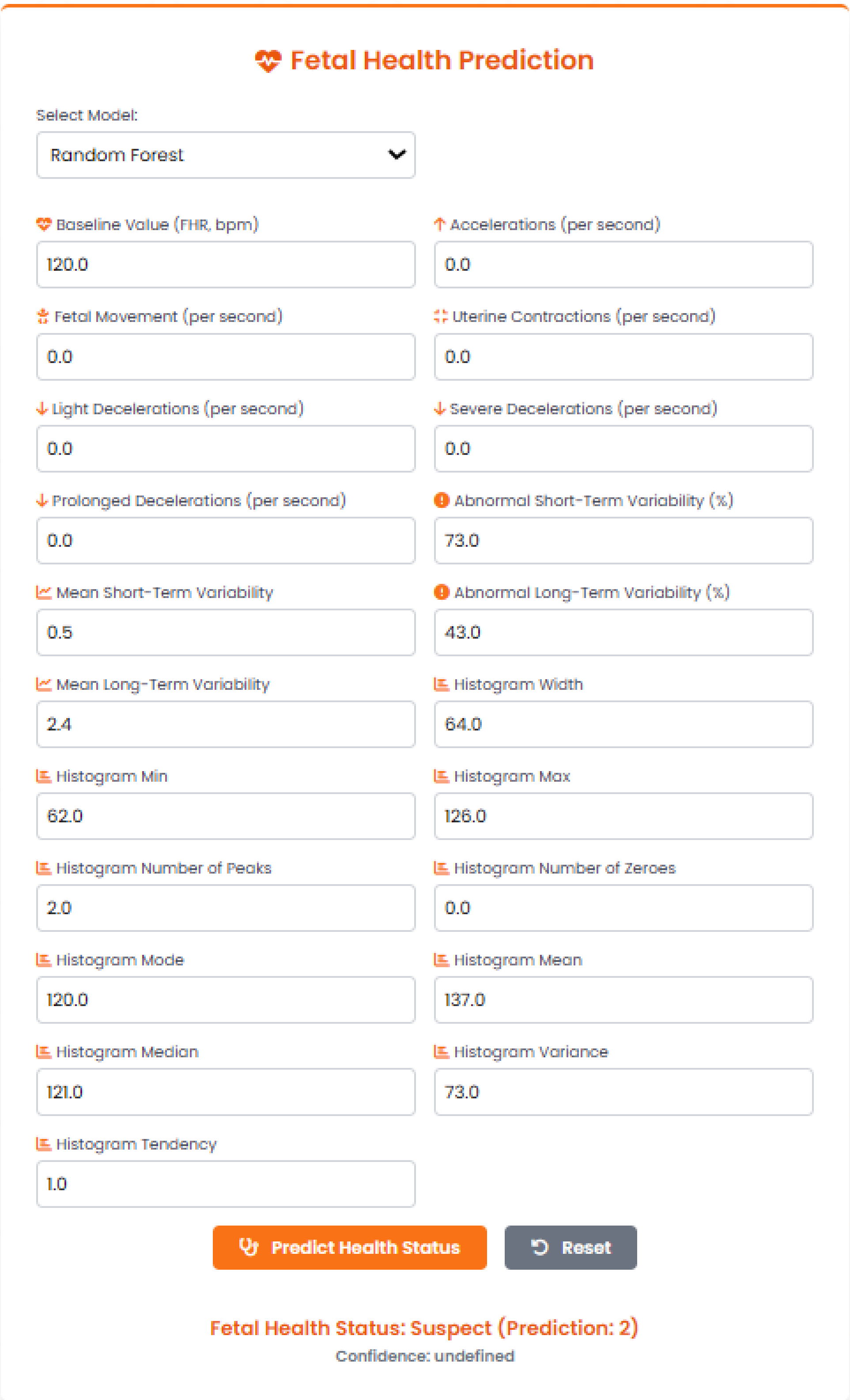
6 Results

6.1 Output Description

The Flask application (`app.py`) provides:

- `index.html`: Form for inputting 21 CTG features.
- `result.html`: Displays prediction (e.g., “Normal”), numeric value (1, 2, 3), and message (e.g., “Consult a healthcare provider”).
- API: Returns JSON with `fetal_health` prediction.

Example: Input `baseline value=120` yields “Normal (1)” with “Normal fetal health” message.


Fetal Health Prediction
Select Model:
Random Forest
Baseline Value (FHR, bpm)
120.0
Accelerations (per second)
0.0
Fetal Movement (per second)
0.0
Uterine Contractions (per second)
0.0
Light Decelerations (per second)
0.0
Severe Decelerations (per second)
0.0
Prolonged Decelerations (per second)
0.0
Abnormal Short-Term Variability (%)
73.0
Mean Short-Term Variability
0.5
Abnormal Long-Term Variability (%)
43.0
Mean Long-Term Variability
2.4
Histogram Width
64.0
Histogram Min
62.0
Histogram Max
126.0
Histogram Number of Peaks
2.0
Histogram Number of Zeroes
0.0
Histogram Mode
120.0
Histogram Mean
137.0
Histogram Median
121.0
Histogram Variance
73.0
Histogram Tendency
1.0
Predict Health Status
Reset
Fetal Health Status: Suspect (Prediction: 2)
Confidence: undefined

7 Advantages & Disadvantages

7.1 Advantages

- High accuracy (~90%) for reliable fetal health predictions.
- User-friendly web interface accessible to non-experts.
- API support for medical system integration.
- Effective handling of class imbalance via XGBoost.

7.2 Disadvantages

- Limited to three health categories, potentially missing nuanced cases.
- Requires valid numerical inputs without robust validation.
- XGBoost offers limited interpretability compared to simpler models.

8 Conclusion

FetalAI successfully delivers an automated fetal health classification system, achieving high accuracy and usability. It streamlines prenatal care, reduces specialist workload, and supports timely interventions, with potential for clinical and research applications.

9 Future Scope

- Add SHAP or LIME for model interpretability.
- Enhance input validation in the Flask app.
- Expand dataset with diverse CTG scenarios.
- Deploy on a cloud platform using Docker.

10 Appendix

10.1 Source Code

Files: `app.py`, `train_model.py`, `index.html`, `result.html`, `fetalai_model.pkl`, `scaler.pkl`, `requirements.txt`.

10.2 Project Resources

- **GitHub Repository**: sakshiipatil/Fetal-Health
- **Project Demo**: https://drive.google.com/file/d/1hCvAlXMXQ1mXDOtuQS3mC4qVcxBwBYZi/view?usp=drivesdk