FetalAI: Using Machine Learning To Predict And Monitor Fetal Health

Prepared For

Smart-Internz

Applied Data

Science

By

Neha Ganesh Kurud

D Y Patil Agriculture and Technical University,
Talsande

On

24 July 2025

1. Introduction: Fetal health monitoring is essential for identifying any abnormal patterns during pregnancy that may impact the fetus. This project, **FetalAI**, uses machine learning models to analyze fetal health based on cardiotocography (CTG) data. The system classifies fetal health into **Normal**, **Suspect**, and **Pathological** categories using both manual input and CSV file predictions.

2. Dataset Description

- **Source**: UCI Machine Learning Repository
- **File**: fetal_health.csv
- **Total Records**: ~2,100
- **Target Variable**: fetal_health (1=Normal, 2=Suspect, 3=Pathological)
- **Features**: 21 numerical measurements derived from CTG exams.

Key Features:

Feature Name	Description
baseline value	Baseline fetal heart rate (FHR)
accelerations	Accelerations per second
fetal_movement	Fetal movements per second
uterine_contractions	Uterine contractions per second
light_decelerations	Light decelerations per second
severe_decelerations	Severe decelerations per second
prolongued_decelerations	Prolonged decelerations per second
abnormal_short_term_variability	Abnormal STV duration
mean_value_of_short_term_variability	Mean STV value
percentage_of_time_with_abnormal_long_term_variability	Abnormal LTV % time
mean_value_of_long_term_variability	Mean LTV value
histogram_width	Histogram width
histogram_min	Minimum histogram value
histogram_max	Maximum histogram value
histogram_number_of_peaks	Number of peaks
histogram_number_of_zeroes	Number of zero crossings
histogram_mode	
histogram_mean	Mean histogram value
histogram_median	Median histogram value
histogram_variance	Variance in histogram
histogram_tendency	Histogram tendency direction

3. Data Preprocessing

- **Missing Values**: Removed using dropna().
- **Target Split**:
- X = df.drop("fetal_health", axis=1)
- y = df["fetal_health"]
- **Balancing**: Used **SMOTE** to handle class imbalance.
- **Scaling**: Applied **StandardScaler** for feature normalization.
- **Split**: Train-Test split (80-20 ratio).

4. Machine Learning Models

- Random Forest
- Decision Tree
- Logistic Regression
- K-Nearest Neighbors (KNN)

5. Best Model

- **Selected Model**: Based on highest test accuracy.
- **Model Saved**: Using joblib.dump(model, "model.pkl")
- **Scaler Saved**: As scaler.pkl

6. Web Interface (Flask App) Routes:

- / → **Home Page**: Input form with manual input and baseline dropdown.
- /predict → **Prediction Endpoint**:
  - Accepts uploaded .csv file
  - Accepts form-based manual input

Functionality:

- Automatically scales input using scaler.pkl
- Predicts using model.pkl
- Maps output to class labels:
- label_map = {1: "Normal", 2: "Suspect", 3: "Pathological"}

7. Baseline Samples 3 predefined cases for easy testing:

Sample	Type
Healthy Sample	Normal condition values
Suspect Sample	Mildly abnormal
Pathological Sample	Critical condition

8. File Structure

```
fetal_health_project/
│
├── fetal_health.csv
├── model.pkl
├── scaler.pkl
├── model_accuracy_comparison.png
│
├── app.py
├── templates/
│   ├── index.html
│   └── result.html
│
├── static/
│   └── style.css (optional)
├── uploads/
│   └── (uploaded .csv files)
```

9. Code & results

training_fetal_health_model

```
# 🛍️ Import Required Libraries
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import classification_report, accuracy_score
from sklearn.ensemble import RandomForestClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.neighbors import KNeighborsClassifier
from imblearn.over_sampling import SMOTE
import joblib
import matplotlib.pyplot as plt

# 📥 Load Dataset
df = pd.read_csv("fetal_health.csv")

#  Step 1: Handle Missing Values
df.dropna(inplace=True)

#  Step 2: Feature and Target Separation
X = df.drop("fetal_health", axis=1)
y = df["fetal_health"]

# ⚖️ Step 3: Balance Dataset using SMOTE
smote = SMOTE(random_state=42)
```

```
X_res, y_res = smote.fit_resample(X, y)

#  Step 4: Train-Test Split
X_train, X_test, y_train, y_test = train_test_split(X_res, y_res,
test_size=0.2, random_state=42)

# 📏 Step 5: Feature Scaling
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)

# 💾 Save scaler for future use
joblib.dump(scaler, "scaler.pkl")

#  Step 6: Train Models models
= {
    "Random Forest": RandomForestClassifier(),
    "Decision Tree": DecisionTreeClassifier(),
    "Logistic Regression": LogisticRegression(),
    "KNN": KNeighborsClassifier()
}

# 📊 Step 7: Train & Evaluate Models
best_model = None
best_accuracy = 0
results = {}

for name, model in models.items():
    model.fit(X_train, y_train)
    y_pred = model.predict(X_test)
    acc = accuracy_score(y_test, y_pred)
    results[name] = acc
    print(f"\n{name} Accuracy: {acc}")
    print(classification_report(y_test, y_pred))

    if acc > best_accuracy:
        best_accuracy = acc
        best_model = model

# 💾 Step 8: Save the best model
joblib.dump(best_model, "model.pkl")
print(f"\n✅ Best Model Saved: {type(best_model).__name__} with Accuracy:
{best_accuracy:.2f}")

# 📈 Step 9: Plot Accuracy Comparison
plt.figure(figsize=(10,6))
plt.bar(results.keys(), results.values(), color='skyblue')
plt.title("Model Accuracy Comparison")
```

```
plt.ylabel("Accuracy")
plt.xlabel("Models")
plt.ylim(0, 1)
plt.grid(axis='y', linestyle='--')
plt.tight_layout()
plt.savefig("model_accuracy_comparison.png")
plt.show()
```

app.py

```
from flask import Flask, render_template, request
import pandas as pd
import numpy as np
import joblib
import os

app = Flask(__name__)
UPLOAD_FOLDER = "uploads"
os.makedirs(UPLOAD_FOLDER, exist_ok=True)

# Load model and scaler
model = joblib.load("model.pkl")
scaler = joblib.load("scaler.pkl")
label_map = {1: "Normal", 2: "Suspect", 3: "Pathological"}

# Feature Names
feature_names = [
    'baseline value', 'accelerations', 'fetal_movement',
'uterine_contractions',
    'light_decelerations', 'severe_decelerations', 'prolongued_decelerations',
    'abnormal_short_term_variability', 'mean_value_of_short_term_variability',
    'percentage_of_time_with_abnormal_long_term_variability',
    'mean_value_of_long_term_variability', 'histogram_width', 'histogram_min',
    'histogram_max', 'histogram_number_of_peaks',
'histogram_number_of_zeroes',
    'histogram_mode', 'histogram_mean', 'histogram_median',
    'histogram_variance', 'histogram_tendency'
]

# Example Baseline Samples
baseline_samples = {
    "Healthy Sample": [120.0, 0.005, 0.002, 0.0, 0.0, 0.0, 0.0, 0.0, 0.5,
                       0.0, 0.5, 50, 60, 160, 1, 0, 150, 140, 140, 10, 1],
     "Suspect Sample": [100.0, 0.001, 0.004, 0.002, 0.01, 0.0, 0.0, 1.0, 0.3,
                        10.0, 0.6, 30, 50, 140, 3, 0, 120, 110, 105, 12, -1],
```

```
    "Pathological Sample": [80.0, 0.0, 0.001, 0.003, 0.02, 0.01, 0.01, 2.0,
0.2,
                            20.0, 0.4, 40, 30, 110, 5, 1, 100, 90, 85, 20, -1]
}

@app.route('/')
def index():
    return render_template("index.html", feature_names=feature_names,
baselines=baseline_samples)

@app.route('/predict', methods=['POST'])
def predict():
    # If CSV is uploaded
    if 'csv_file' in request.files and request.files['csv_file'].filename !=
'':
        file = request.files['csv_file']
        filepath = os.path.join(UPLOAD_FOLDER, file.filename)
        file.save(filepath)
        df = pd.read_csv(filepath)

        if "fetal_health" in df.columns:
            df = df.drop("fetal_health", axis=1)

        scaled = scaler.transform(df)
        predictions = model.predict(scaled)
        df["Prediction"] = [label_map.get(p, "Unknown") for p in predictions]

        return render_template("result.html",
tables=[df.to_html(classes='table table-sm table-striped', index=False)])

    # Manual Input Prediction
    try:
        input_values = []
        for f in feature_names:
            val = request.form.get(f)
            if val is None or val.strip() == '':
                return f"❌ Missing or invalid input for field: {f}"
            try:
                input_values.append(float(val))
            except ValueError:
                return f"❌ Invalid numeric value for field: {f} → {val}"

        X_input = np.array(input_values).reshape(1, -1)
        X_scaled = scaler.transform(X_input)
        prediction = model.predict(X_scaled)[0]
        label = label_map.get(prediction, "Unknown")

        result_df = pd.DataFrame([input_values], columns=feature_names)
```

```
        result_df["Prediction"] = label

        return render_template("result.html",
tables=[result_df.to_html(classes='table table-sm table-striped',
index=False)])
    except Exception as e:
        return f"❌ Error processing input: {e}"

if __name__== '__main__':
    app.run(debug=True)
```

index.html

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<title>Fetal Health Predictor</title>
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css"
rel="stylesheet">

<style>
  body {
    font-family: 'Segoe UI', sans-serif;
    background: linear-gradient(120deg, #e0f7fa, #e1bee7);
    min-height: 100vh;
  }
  /* Navbar */
  .navbar {
    backdrop-filter: blur(12px);
    background: rgba(255, 255, 255, 0.3) !important;
    border-bottom: 1px solid rgba(255,255,255,0.2);
  }
  .navbar-brand {
    font-weight: 700;
    color: #4a148c !important;
  }
  /* Sidebar */
  .sidebar {
    background: rgba(255,255,255,0.3);
    backdrop-filter: blur(10px);
    border-right: 1px solid rgba(255,255,255,0.2);
    padding-top: 1rem;
    height: 100%;
    position: fixed;
    top: 56px;
    left: 0;
    width: 220px;
    transition: all 0.3s ease;
```

```
<style>
  }
  .sidebar a {
    display: block;
    padding: 12px 20px;
    color: #4a148c;
    text-decoration: none;
    font-weight: 500;
    border-radius: 8px;
    margin: 5px 10px;
  }
  .sidebar a.active,
  .sidebar a:hover {
    background-color: #4a148c;
    color: white;
  }
  /* Main content */
  .main-content {
    margin-left: 220px;
    padding: 2rem;
    transition: margin-left 0.3s ease;
  }
  @media (max-width: 768px) {
    .sidebar {
      transform: translateX(-100%);
    }
    .sidebar.show {
      transform: translateX(0);
    }
    .main-content {
      margin-left: 0;
    }
  }
  /* Cards */
  .glass-card {
    background: rgba(255,255,255,0.4);
<style>
  .glass-card {
    background: rgba(255,255,255,0.4);
    backdrop-filter: blur(12px);
    border-radius: 16px;
    padding: 2rem;
    box-shadow: 0px 8px 20px rgba(0,0,0,0.1);
    margin-bottom: 2rem;
    animation: fadeIn 0.5s ease-in-out;
  }
  /* Animation */
  @keyframes fadeIn {
    from {opacity: 0; transform: translateY(10px);}
    to {opacity: 1; transform: translateY(0);}
  }
  /* Content tabs */
  .content-tab {
    display: none;
  }
  .content-tab.active {
    display: block;
  }
</style>
</head>
<body>

<!-- Navbar -->
<nav class="navbar navbar-expand-lg fixed-top">
  <div class="container-fluid">
    <button class="btn btn-outline-primary d-lg-none me-2" id="menuToggle">☰</button>
    <a class="navbar-brand" href="#">FetalAI</a>
    <div class="collapse navbar-collapse justify-content-end">
      <a class="btn btn-primary" onclick="showTab('contactTab')">📞 Contact Us</a>
    </div>
  </div>
</nav>
```

```
<!-- Sidebar -->
<div class="sidebar" id="sidebar">
  <a href="#" class="active" onclick="showTab('homeTab')">🏠 Home</a>
  <a href="#" onclick="showTab('uploadTab')">📄 Upload CSV</a>
  <a href="#" onclick="showTab('manualTab')">✍️ Manual Input</a>
  <a href="#" onclick="showTab('contactTab')">📞 Contact Us</a>
</div>

<!-- Main Content -->
<div class="main-content">

  <!-- Home Tab -->
  <div id="homeTab" class="content-tab active">
    <div class="glass-card text-center mt-5">
      <h1 class="mb-4">Welcome to Fetal Health Prediction</h1>
      <img src="https://th.bing.com/th/id/OIP.woZ5jn9-uoVZzGDjjguUiQHaFN?w=253&h=180&c=7&r=0&o=7&dpr=1.3&pid=1.7&rm=3"
           alt="Fetal Health" class="img-fluid rounded shadow" style="max-height: 400px;">
    </div>
  </div>

  <!-- Upload CSV Tab -->
  <div id="uploadTab" class="content-tab">
    <div class="glass-card mt-5">
      <h4>📄 Upload CSV</h4>
      <form action="/predict" method="post" enctype="multipart/form-data">
        <div class="mb-3">
          <input type="file" name="csv_file" accept=".csv" class="form-control" required>
        </div>
        <button type="submit" class="btn btn-primary">Predict from CSV</button>
      </form>
    </div>
  </div>

  <!-- Manual Input Tab -->
  <div id="manualTab" class="content-tab">
    <div class="glass-card mt-5">
      <h4>✍️ Manual Input</h4>
```

```
    <h4>✍️ Manual Input</h4>
    <div class="mb-3">
      <label class="form-label">Select Baseline</label>
      <select id="baselineSelect" class="form-select" onchange="fillBaseline(this.value)">
        <option value="">-- Choose --</option>
        {% for label, values in baselines.items() %}
          <option value="{{ values|join(',') }}">{{ label }}</option>
        {% endfor %}
      </select>
    </div>
    <form action="/predict" method="post" id="manualForm">
      <div class="row">
        {% for field in feature_names %}
          <div class="col-md-4 mb-3">
            <label class="form-label">{{ field.replace('_', ' ').capitalize() }}</label>
            <input type="number" step="any" name="{{ field }}" id="{{ field }}" class="form-control" required>
          </div>
        {% endfor %}
      </div>
      <button type="submit" class="btn btn-success">Predict Manually</button>
    </form>
  </div>
</div>

<!-- Contact Tab -->
<div id="contactTab" class="content-tab">
  <div class="glass-card mt-5">
    <h4>📞 Contact Us</h4>
    <p>If you have any questions or feedback, feel free to reach out:</p>
    <ul>
      <li>Email: support@fetalai.com</li>
      <li>Phone: +91-9876543210</li>
    </ul>
```

```
<script>
function showTab(tabId) {
  document.querySelectorAll('.content-tab').forEach(tab => tab.classList.remove('active'));
  document.getElementById(tabId).classList.add('active');

  document.querySelectorAll('.sidebar a').forEach(link => link.classList.remove('active'));
  const activeLink = [...document.querySelectorAll('.sidebar a')].find(link => link.getAttribute('onclick'
  if (activeLink) activeLink.classList.add('active');
}

function fillBaseline(valuesStr) {
  if (!valuesStr) return;
  const values = valuesStr.split(',');
  const fields = {{ feature_names|tojson }};
  fields.forEach((name, i) => {
    const input = document.getElementById(name);
    if (input) input.value = values[i];
  });
}

document.getElementById('menuToggle').addEventListener('click', () => {
  document.getElementById('sidebar').classList.toggle('show');
});
</script>
<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/js/bootstrap.bundle.min.js"></script>
</body>
</html>
```

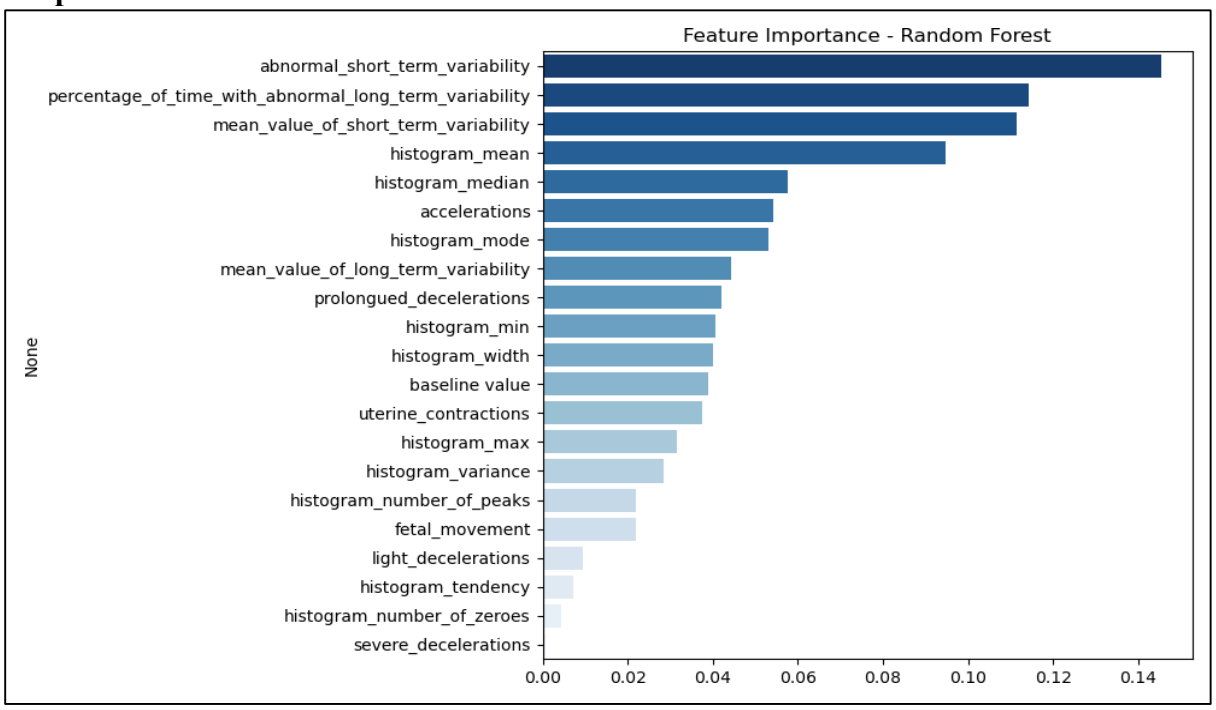
Result.html

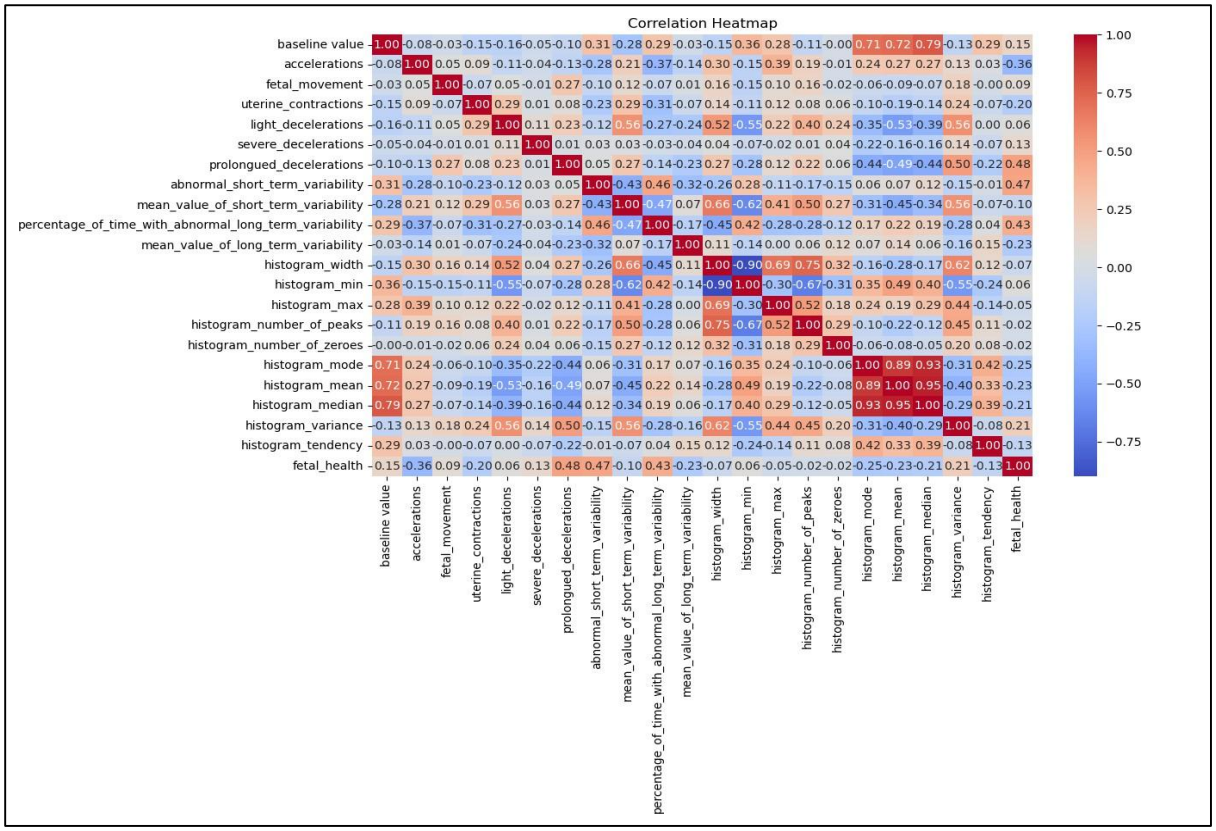
```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Prediction Result</title>
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css" rel="stylesheet">

  <style>
    body {
      font-family: 'Segoe UI', sans-serif;
      background: linear-gradient(120deg, #e0f7fa, #e1bee7);
      min-height: 100vh;
      display: flex;
      justify-content: center;
      align-items: center;
      padding: 20px;
    }
    .glass-card {
      background: rgba(255,255,255,0.4);
      backdrop-filter: blur(12px);
      border-radius: 16px;
      padding: 2rem;
      box-shadow: 0px 8px 20px rgba(0,0,0,0.1);
      width: 100%;
      max-width: 900px;
      animation: fadeIn 0.5s ease-in-out;
    }
    table {
      background-color: white;
      border-radius: 8px;
      overflow: hidden;
    }
    @keyframes fadeIn {
      from {opacity: 0; transform: translateY(10px);}
      to {opacity: 1; transform: translateY(0);}
    }
```

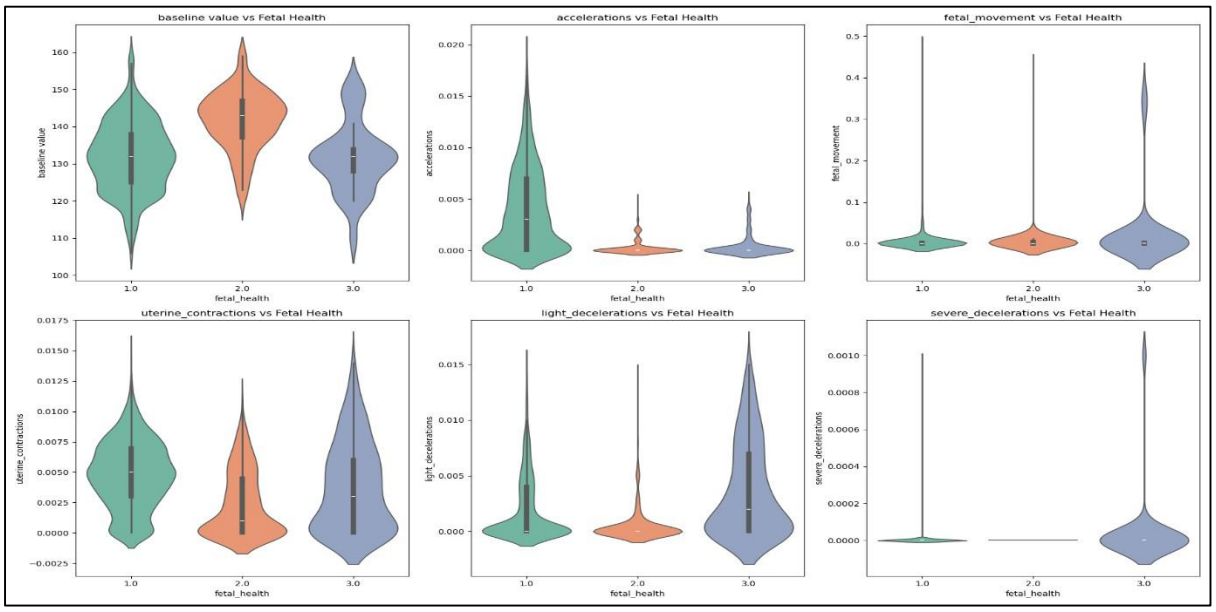
```
<head>
  <style>
    }
  </style>
</head>
<body>

<div class="glass-card">
  <h2 class="mb-4 text-primary">📊 Prediction Results</h2>
  <a href="/" class="btn btn-secondary mb-3">⬅ Back</a>

  {% for table in tables %}
    <div class="table-responsive">
      {{ table | safe }}
    </div>
  {% endfor %}
</div>

</body>
</html>
```

SMARTBRIDGE
Let's Bridge the Gap
a Veranda Enterprise

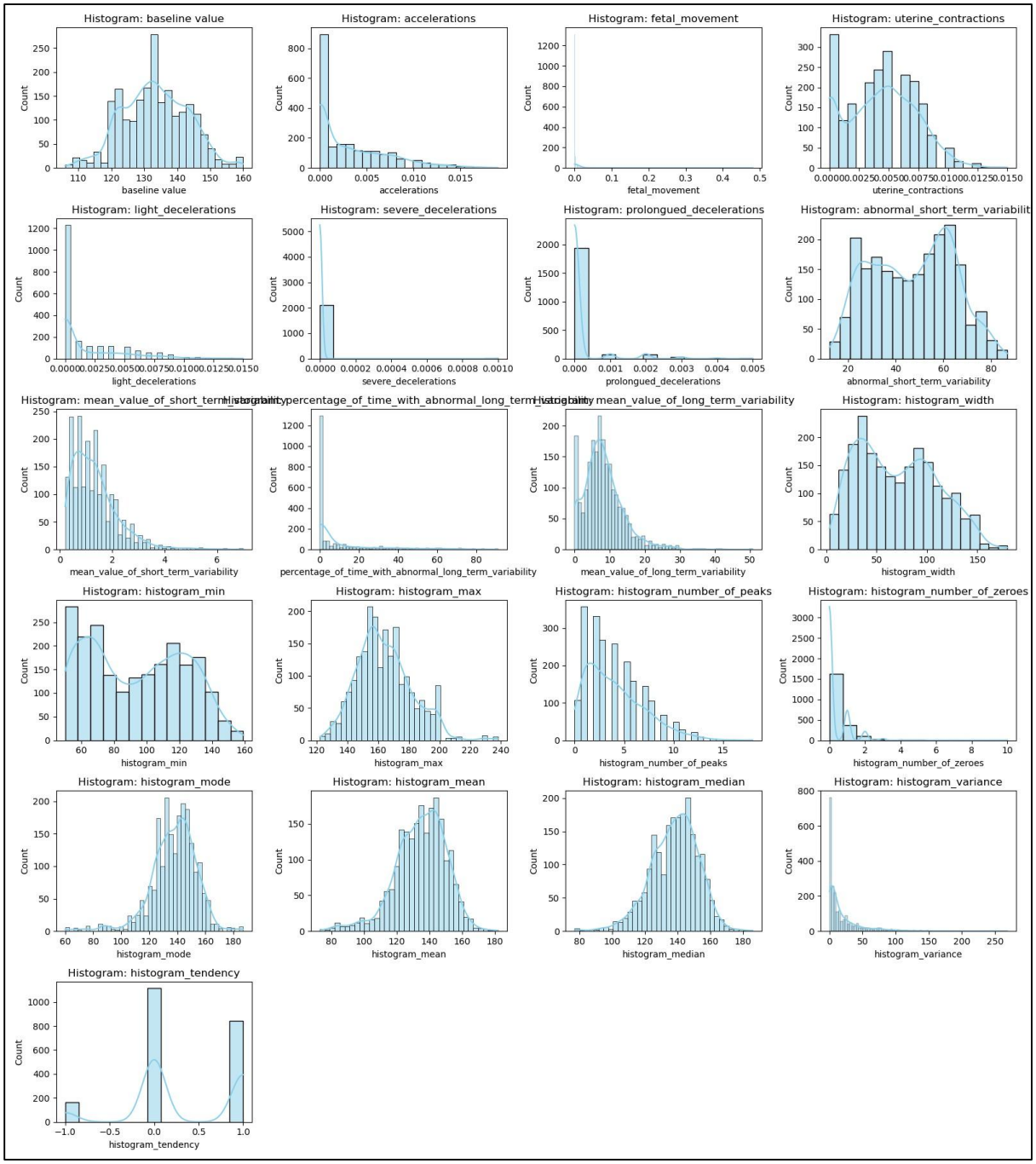
Outputs:

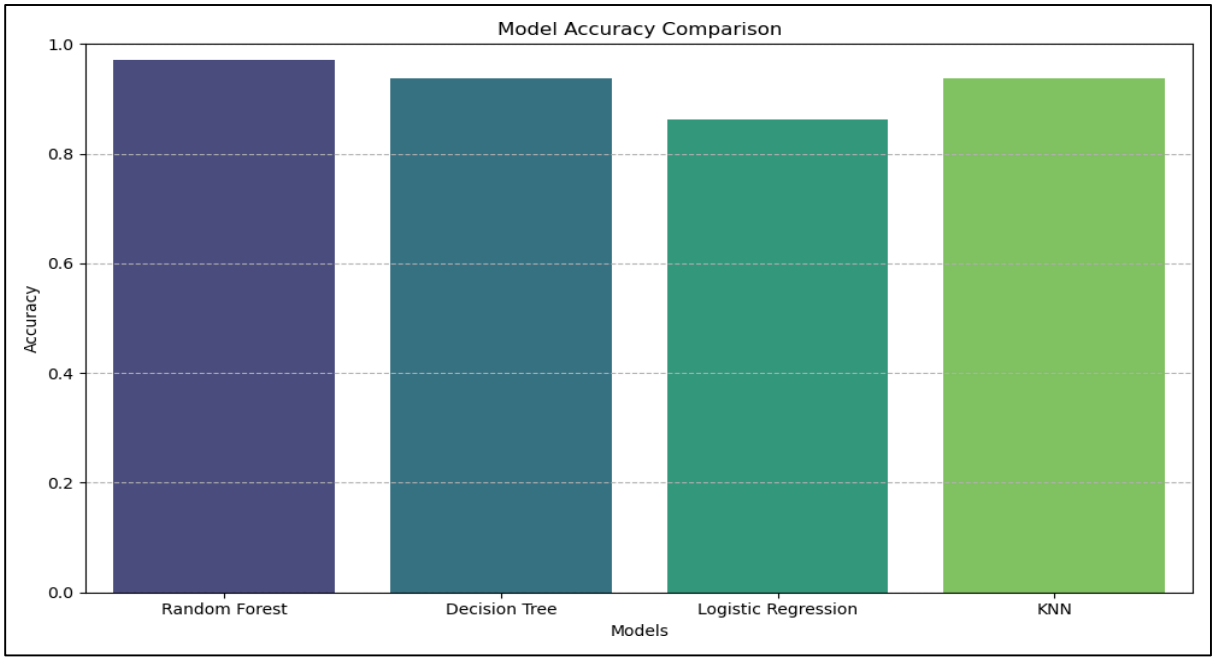


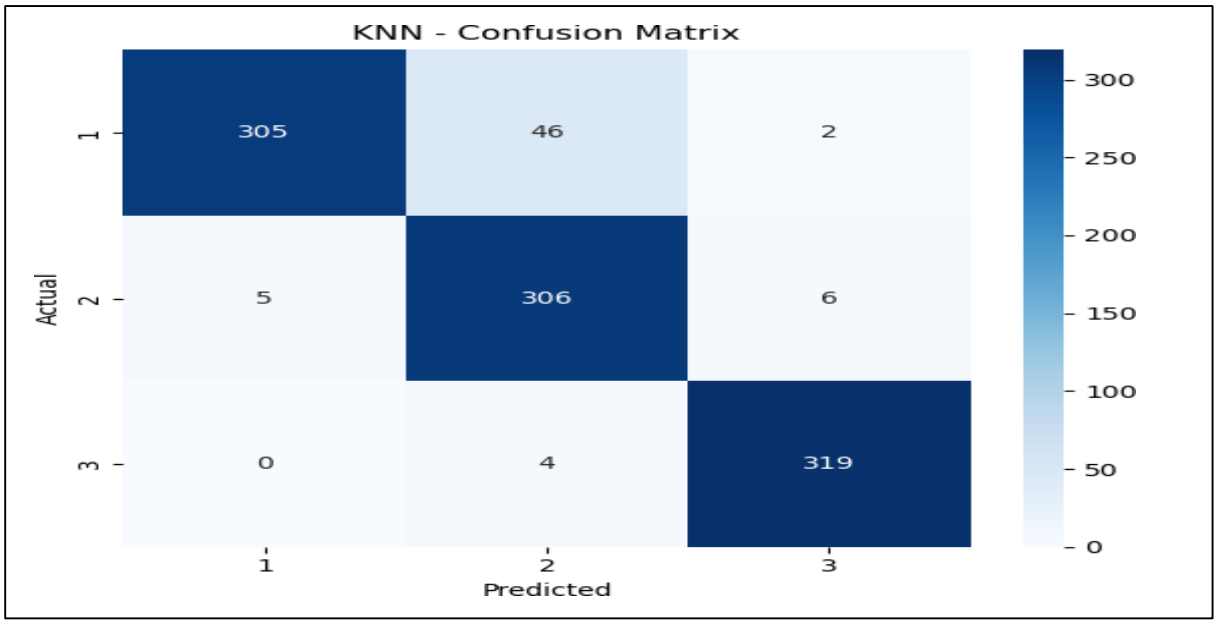


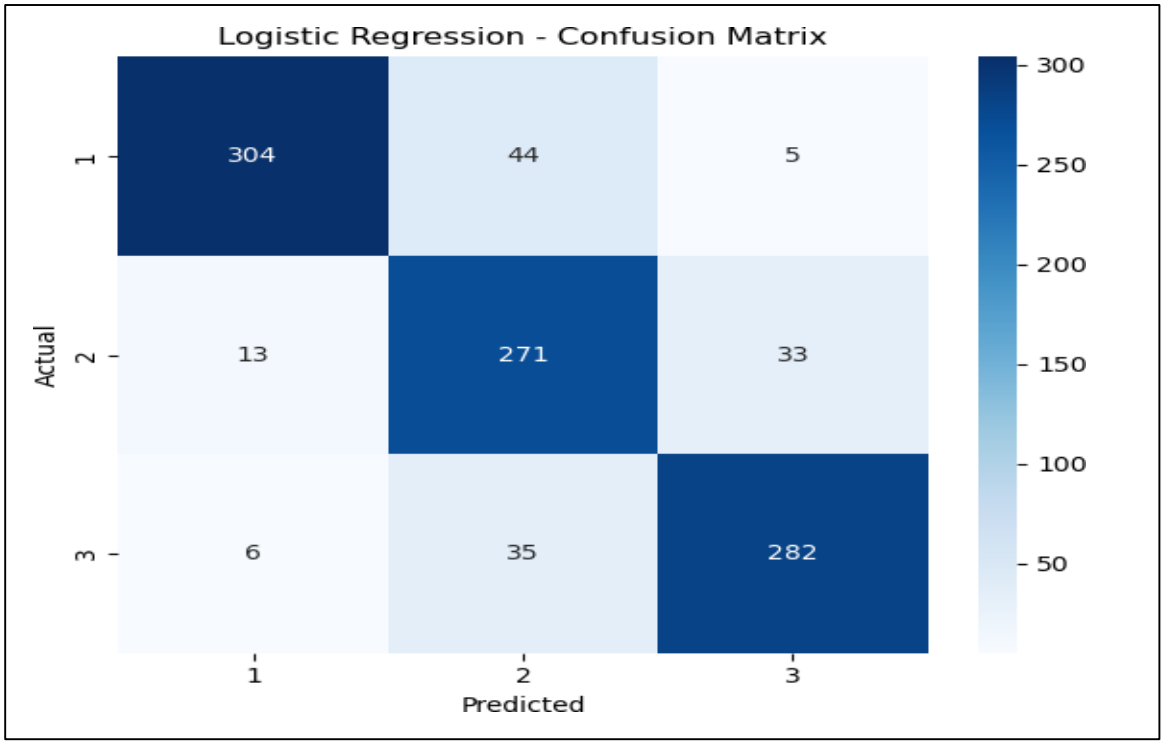

baseline value vs Fetal Health
accelerations vs Fetal Health
fetal_movement vs Fetal Health
uterine_contractions vs Fetal Health
light_decelerations vs Fetal Health
severe_decelerations vs Fetal Health
fetal_health

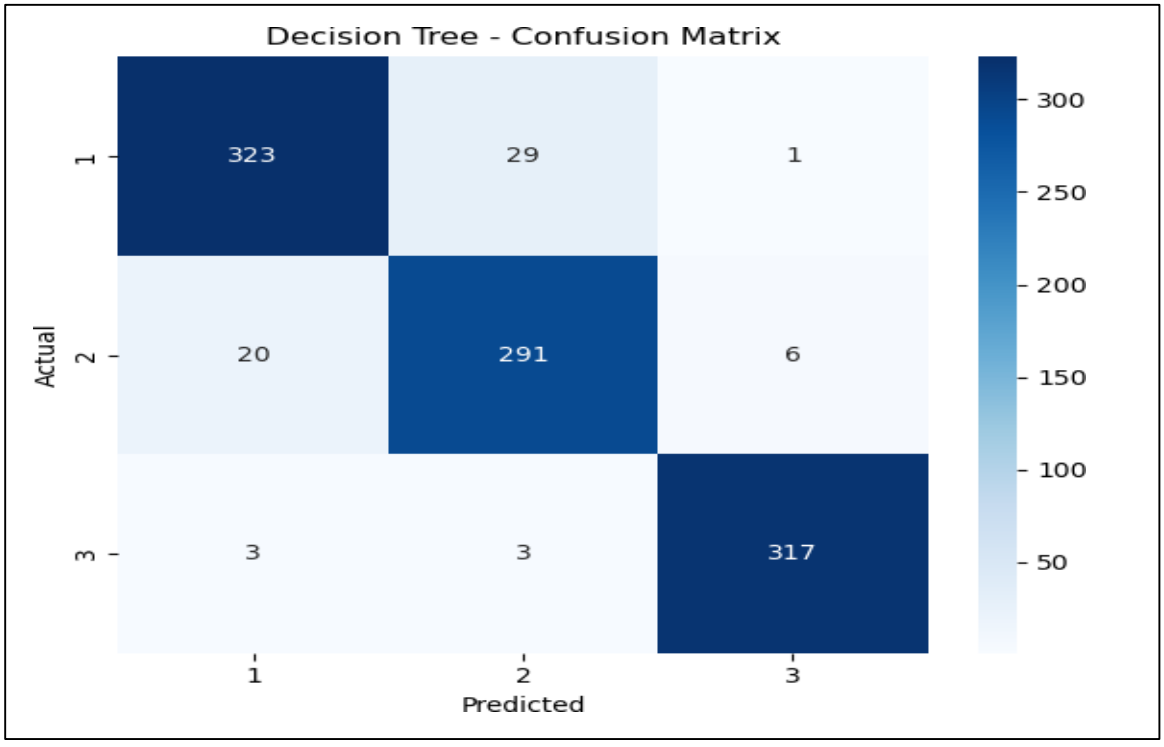

Boxplot: baseline value
Boxplot: accelerations
Boxplot: fetal_movement
Boxplot: uterine_contractions
Boxplot: light_decelerations
Boxplot: severe_decelerations
Boxplot: prolongued_decelerations
Boxplot: abnormal_short_term_variability
Boxplot: mean_value_of_short_term_variability
Boxplot: percentage_of_time_with_abnormal_long_term_variability
Boxplot: mean_value_of_long_term_variability
Boxplot: histogram_width
Boxplot: histogram_min
Boxplot: histogram_max
Boxplot: histogram_number_of_peaks
Boxplot: histogram_number_of_zeroes
Boxplot: histogram_mode
Boxplot: histogram_mean
Boxplot: histogram_median
Boxplot: histogram_variance
Boxplot: histogram_tendency

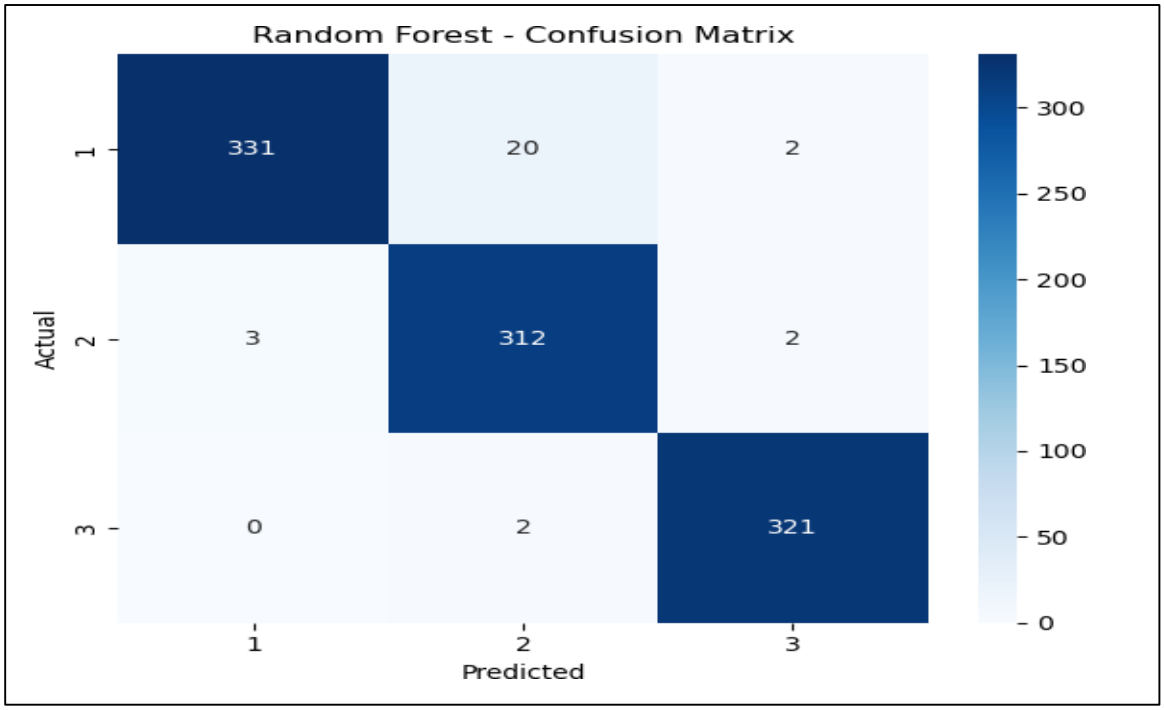

Histogram: baseline value
baseline value
Histogram: accelerations
accelerations
Histogram: fetal_movement
fetal_movement
Histogram: uterine_contractions
uterine_contractions
Histogram: light_decelerations
light_decelerations
Histogram: severe_decelerations
severe_decelerations
Histogram: prolongued_decelerations
prolongued_decelerations
Histogram: abnormal_short_term_variabilit
abnormal_short_term_variability
Histogram: mean_value_of_short_term_variability
mean_value_of_short_term_variability
Histogram: percentage_of_time_with_abnormal_long_term_variability
percentage_of_time_with_abnormal_long_term_variability
Histogram: mean_value_of_long_term_variability
mean_value_of_long_term_variability
Histogram: histogram_width
histogram_width
Histogram: histogram_min
histogram_min
Histogram: histogram_max
histogram_max
Histogram: histogram_number_of_peaks
histogram_number_of_peaks
Histogram: histogram_number_of_zeroes
histogram_number_of_zeroes
Histogram: histogram_mode
histogram_mode
Histogram: histogram_mean
histogram_mean
Histogram: histogram_median
histogram_median
Histogram: histogram_variance
histogram_variance
Histogram: histogram_tendency
histogram_tendency
Count

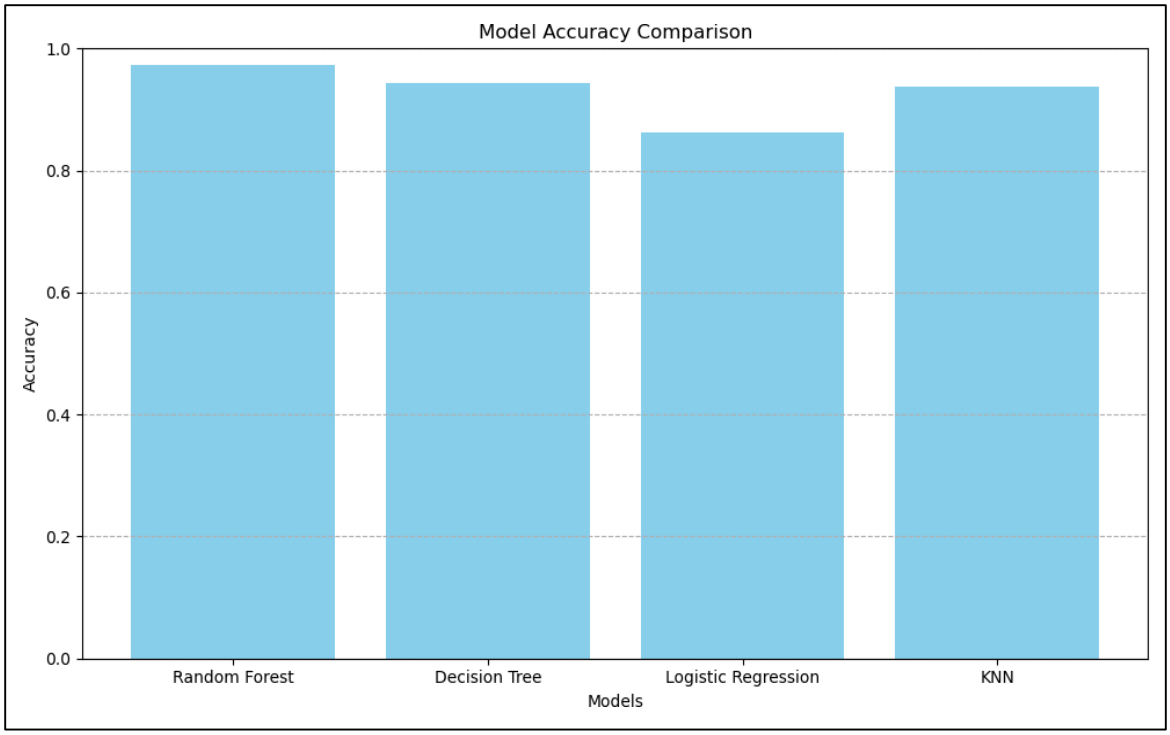

Model Accuracy Comparison
Accuracy
1.0
0.8
0.6
0.4
0.2
0.0
Random Forest
Decision Tree
Logistic Regression
KNN
Models


KNN - Confusion Matrix
Actual
1
2
3
305
46
2
5
306
6
0
4
319
1
2
3
Predicted
300
250
200
150
100
50
0

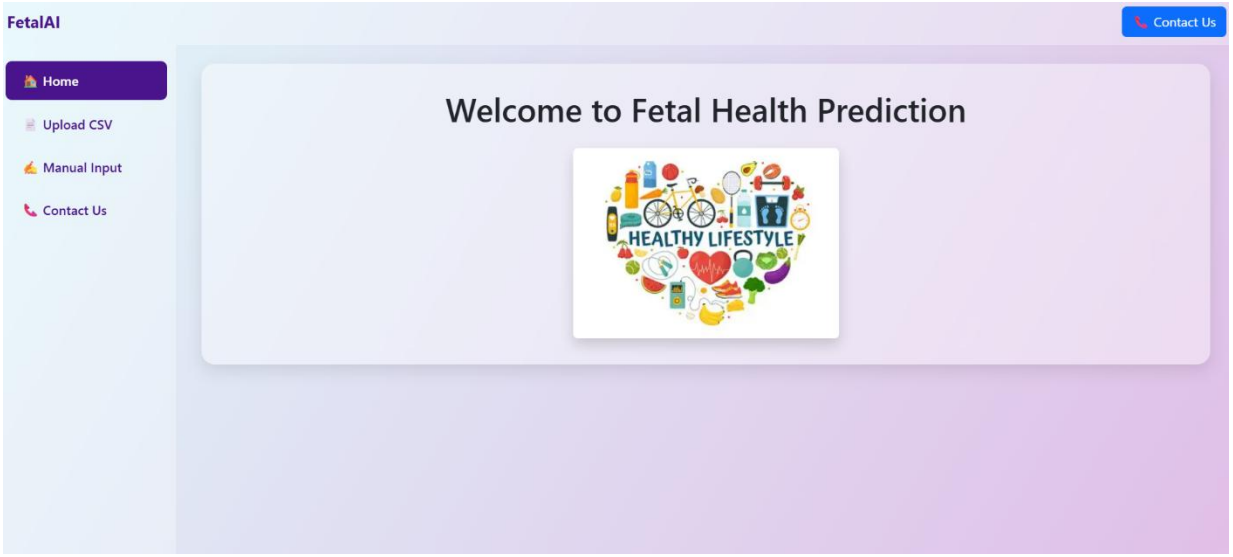

Logistic Regression - Confusion Matrix
304
44
5
13
271
33
6
35
282
Actual
Predicted
1
2
3
300
250
200
150
100
50

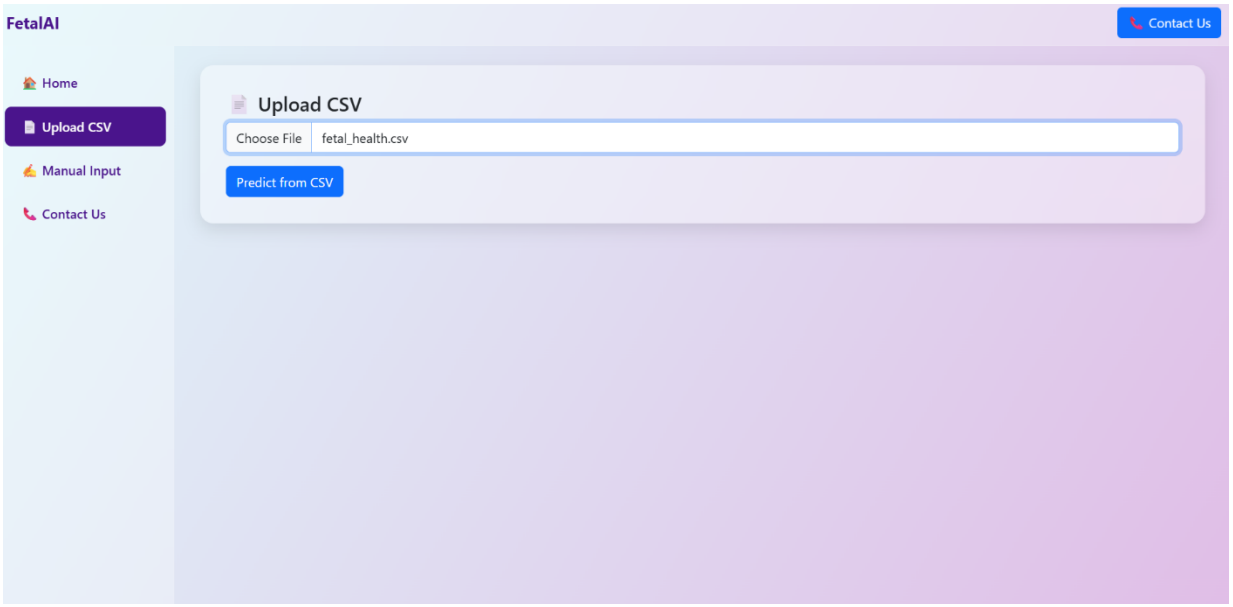

Decision Tree - Confusion Matrix
323
29
1
20
291
6
3
3
317
Actual
Predicted
1
2
3
300
250
200
150
100
50

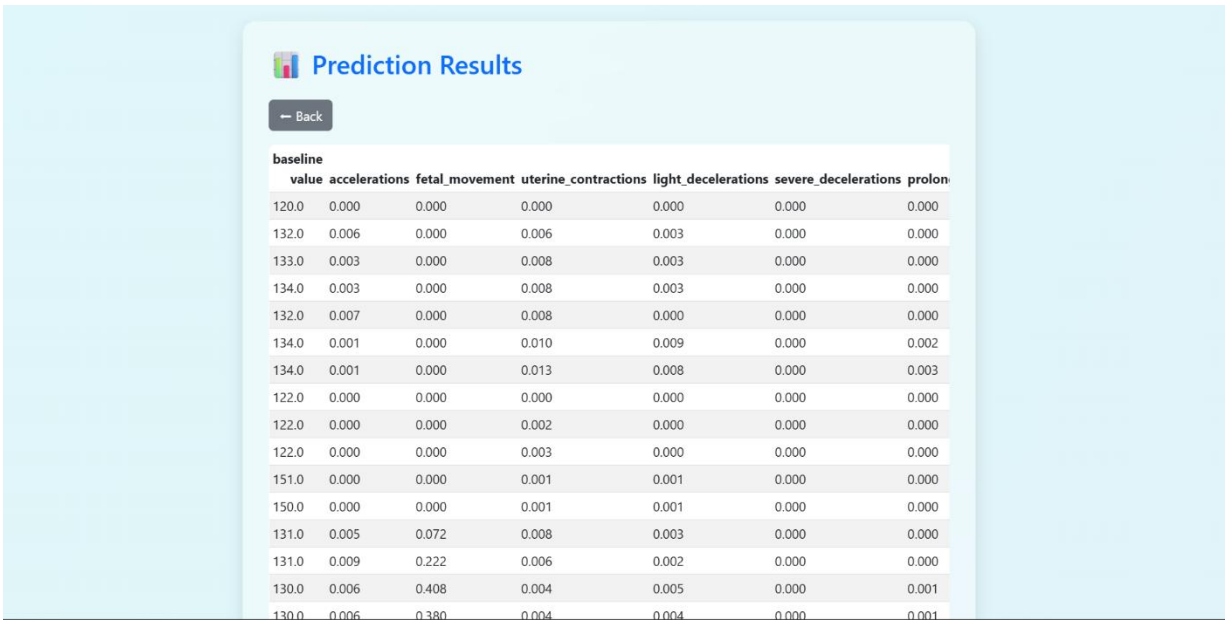

Random Forest - Confusion Matrix
331
20
2
3
312
2
0
2
321
Actual
1
2
3
Predicted
1
2
3
300
250
200
150
100
50
0


Model Accuracy Comparison
Accuracy
1.0
0.8
0.6
0.4
0.2
0.0
Random Forest
Decision Tree
Logistic Regression
KNN
Models

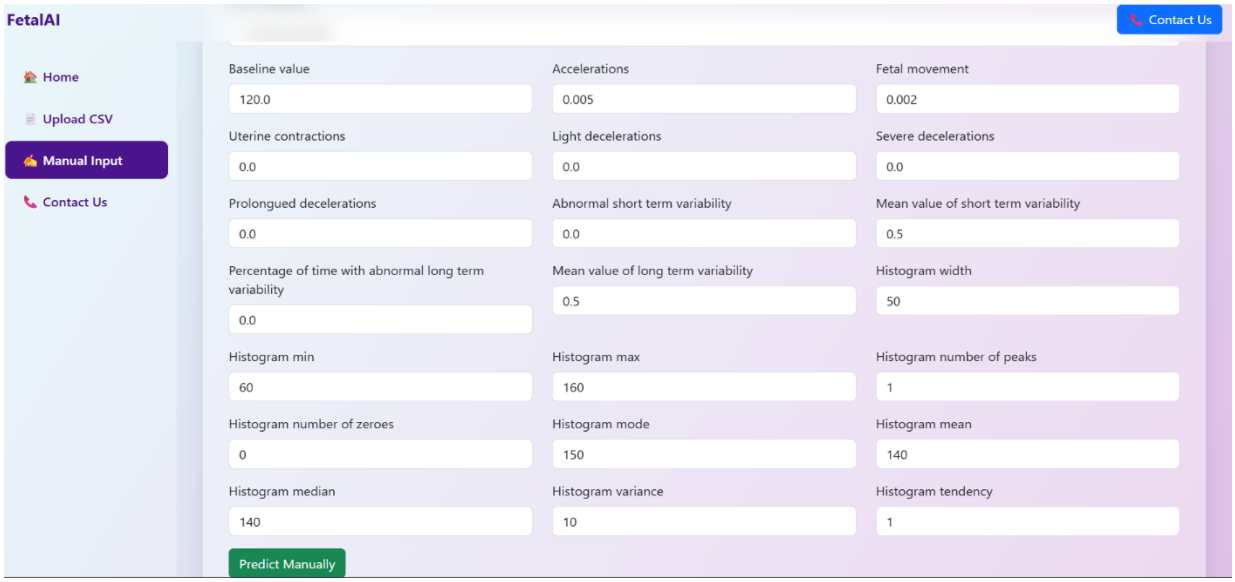
GUI:

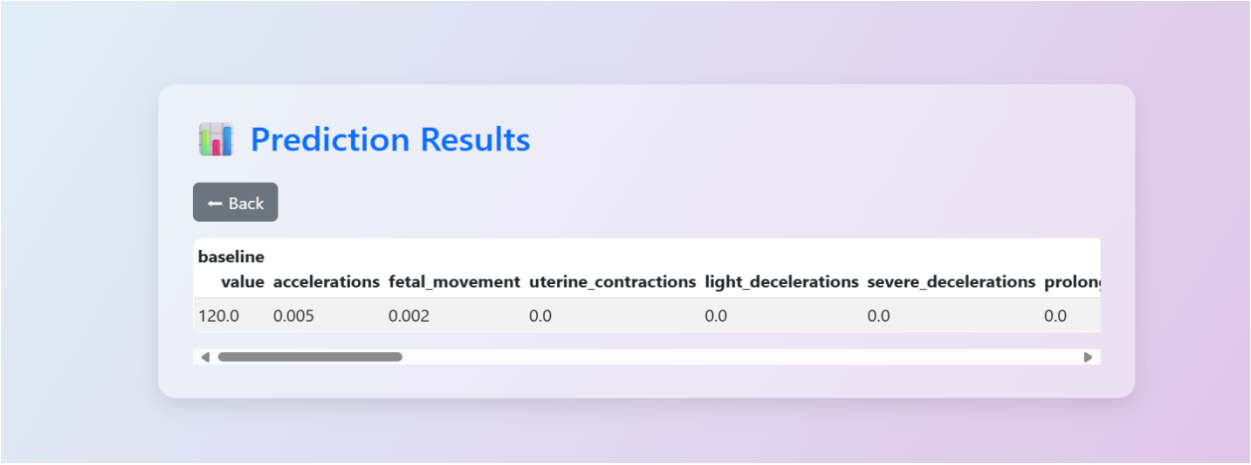


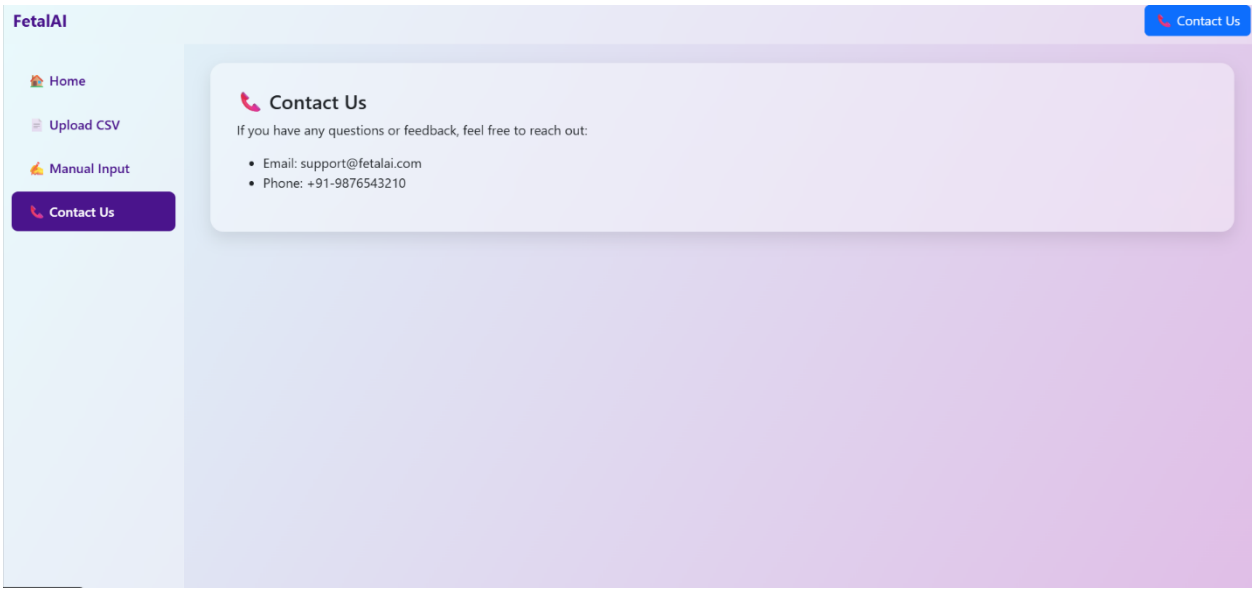




baseline value	accelerations	fetal_movement	uterine_contractions	light_decelerations	severe_decelerations	prolon
120.0	0.000	0.000	0.000	0.000	0.000	0.000
132.0	0.006	0.000	0.006	0.003	0.000	0.000
133.0	0.003	0.000	0.008	0.003	0.000	0.000
134.0	0.003	0.000	0.008	0.003	0.000	0.000
132.0	0.007	0.000	0.008	0.000	0.000	0.000
134.0	0.001	0.000	0.010	0.009	0.000	0.002
134.0	0.001	0.000	0.013	0.008	0.000	0.003
122.0	0.000	0.000	0.000	0.000	0.000	0.000
122.0	0.000	0.000	0.002	0.000	0.000	0.000
122.0	0.000	0.000	0.003	0.000	0.000	0.000
151.0	0.000	0.000	0.001	0.001	0.000	0.000
150.0	0.000	0.000	0.001	0.001	0.000	0.000
131.0	0.005	0.072	0.008	0.003	0.000	0.000
131.0	0.009	0.222	0.006	0.002	0.000	0.000
130.0	0.006	0.408	0.004	0.005	0.000	0.001
130.0	0.006	0.380	0.004	0.004	0.000	0.001


FetalAI
Contact Us
Home
Upload CSV
Manual Input
Contact Us
Baseline value
120.0
Accelerations
0.005
Fetal movement
0.002
Uterine contractions
0.0
Light decelerations
0.0
Severe decelerations
0.0
Prolongued decelerations
0.0
Abnormal short term variability
0.0
Mean value of short term variability
0.5
Percentage of time with abnormal long term variability
0.0
Mean value of long term variability
0.5
Histogram width
50
Histogram min
60
Histogram max
160
Histogram number of peaks
1
Histogram number of zeroes
0
Histogram mode
150
Histogram mean
140
Histogram median
140
Histogram variance
10
Histogram tendency
1
Predict Manually


Prediction Results
Back
baseline value
accelerations
fetal_movement
uterine_contractions
light_decelerations
severe_decelerations
prolon
120.0
0.005
0.002
0.0
0.0
0.0
0.0


FetalAI
Contact Us
Home
Upload CSV
Manual Input
Contact Us
Contact Us
If you have any questions or feedback, feel free to reach out:
Email: support@fetalai.com
Phone: +91-9876543210