SOURCE CODE

HTML:(Index.html)

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Fetal AI Project</title>
    <style>
        body {
            font-family: Arial, sans-serif;
            text-align: center;
            margin: 20px;
        }

        button {
            padding: 10px;
            margin: 10px;
            font-size: 16px;
            cursor: pointer;
        }
    </style>
</head>
<body>

    <h1>Fetal AI Project</h1>

    <button id="prolonged_declarations">Prolonged Declarations</button>
    <button id="histogram_variance">Histogram Variance</button>
    <button id="histogram_mode">Histogram Mode</button>
    <button id="accelerations">Accelerations</button>
</body>
</html>
```

Output.html:

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
```

```html
    <title>Predict and Monitor Fetal Health</title>
    <style>
        body {
            font-family: Arial, sans-serif;
            margin: 0;
            padding: 0;
            background-color: #f4f4f4;
        }

        header {
            background-color: #333;
            color: #fff;
            text-align: center;
            padding: 1em;
        }

        main {
            max-width: 800px;
            margin: 20px auto;
            padding: 20px;
            background-color: #fff;
            box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
            border-radius: 5px;
        }

        img {
            max-width: 100%;
            height: auto;
            display: block;
            margin: 20px auto;
        }
    </style>
</head>
<body>

    <header>
        <h1>Predict and Monitor Fetal Health</h1>
    </header>

    <main>
        <img src="your_image_url.jpg" alt="Fetal Health Image">
        <!-- Add more content as needed -->
    </main>

</body>
</html>
```

Script.js:

```
<script>
        document.getElementById('prolonged_declarations').addEventListener('cl
ick', function() {
            alert('Prolonged Declarations feature clicked');
        });

        document.getElementById('histogram_variance').addEventListener('click'
, function() {

            alert('Histogram Variance feature clicked');
        });

        document.getElementById('histogram_mode').addEventListener('click',
function() {
            alert('Histogram Mode feature clicked');
        });

        document.getElementById('accelerations').addEventListener('click',
function() {
            // Add logic for accelerations feature
            alert('Accelerations feature clicked');
        });
    </script>
```

FetalAI: Using Machine Learning to predict and monitor Fetal Health

In [124]:
```
import numpy as np
import pandas as pd
#pd.set_option('max_columns', None)
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
sns.set_style('darkgrid')
```

In [ ]:
```
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from imblearn.over_sampling import SMOTE
from sklearn.linear_model import LogisticRegression
#from sklearn.neighbors import kNeighborsClassifier
from sklearn.tree import DecisionTreeClassifier
#from sklearn.svm import Linearsvc, SVC
from sklearn.neural_network import MLPClassifier
from sklearn.ensemble import RandomForestClassifier, GradientBoostingClassifie
from sklearn.metrics import confusion_matrix
#from sklearn.metrics import plot_confusion_matrix
from sklearn.metrics import ConfusionMatrixDisplay
import warnings
warnings .filterwarnings(action='ignore')
```

In [ ]:
```
data =pd.read_csv("/content/fetal_health.csv")
```

In [ ]:
```
data.head()
```

Out[12]:

	baseline value	accelerations	fetal_movement	uterine_contractions	light_decelerations	severe_dece
0	120.0	0.000	0.0	0.000	0.000	
1	132.0	0.006	0.0	0.006	0.003	
2	133.0	0.003	0.0	0.008	0.003	
3	134.0	0.003	0.0	0.008	0.003	
4	132.0	0.007	0.0	0.008	0.000	

5 rows × 22 columns

In [ ]:
```
data.shape
```

Out[18]:
```
(2126, 22)
```

In [ ]:
```
data.tail()
```

Out[13]:

	baseline value	accelerations	fetal_movement	uterine_contractions	light_decelerations	severe_d
2121	140.0	0.000	0.000	0.007	0.0	
2122	140.0	0.001	0.000	0.007	0.0	
2123	140.0	0.001	0.000	0.007	0.0	
2124	140.0	0.001	0.000	0.006	0.0	
2125	142.0	0.002	0.002	0.008	0.0	

5 rows × 22 columns

In [ ]:
```
data.describe()
```

Out[14]:

	baseline value	accelerations	fetal_movement	uterine_contractions	light_decelerations	seve
count	2126.000000	2126.000000	2126.000000	2126.000000	2126.000000	
mean	133.303857	0.003178	0.009481	0.004366	0.001889	
std	9.840844	0.003866	0.046666	0.002946	0.002960	
min	106.000000	0.000000	0.000000	0.000000	0.000000	
25%	126.000000	0.000000	0.000000	0.002000	0.000000	
50%	133.000000	0.002000	0.000000	0.004000	0.000000	
75%	140.000000	0.006000	0.003000	0.007000	0.003000	
max	160.000000	0.019000	0.481000	0.015000	0.015000	

8 rows × 22 columns

In [ ]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2126 entries, 0 to 2125
Data columns (total 22 columns):
 #   Column                                                  Non-Null Count
Dtype
---  ------                                                  --------------
-----
 0   baseline value                                          2126 non-null
float64
 1   accelerations                                           2126 non-null
float64
 2   fetal_movement                                          2126 non-null
float64
 3   uterine_contractions                                    2126 non-null
float64
 4   light_decelerations                                     2126 non-null
float64
 5   severe_decelerations                                    2126 non-null
float64
 6   prolongued_decelerations                                2126 non-null
float64
 7   abnormal_short_term_variability                         2126 non-null
float64
 8   mean_value_of_short_term_variability                    2126 non-null
float64
 9   percentage_of_time_with_abnormal_long_term_variability  2126 non-null
float64
 10  mean_value_of_long_term_variability                     2126 non-null
float64
 11  histogram_width                                         2126 non-null
float64
 12  histogram_min                                           2126 non-null
float64
 13  histogram_max                                           2126 non-null
float64
 14  histogram_number_of_peaks                               2126 non-null
float64
 15  histogram_number_of_zeroes                              2126 non-null
float64
 16  histogram_mode                                          2126 non-null
float64
 17  histogram_mean                                          2126 non-null
float64
 18  histogram_median                                        2126 non-null
float64
 19  histogram_variance                                      2126 non-null
float64
 20  histogram_tendency                                      2126 non-null
float64
 21  fetal_health                                            2126 non-null
float64
dtypes: float64(22)
memory usage: 365.5 KB
```

In [ ]:
```
data.isnull().sum()
```

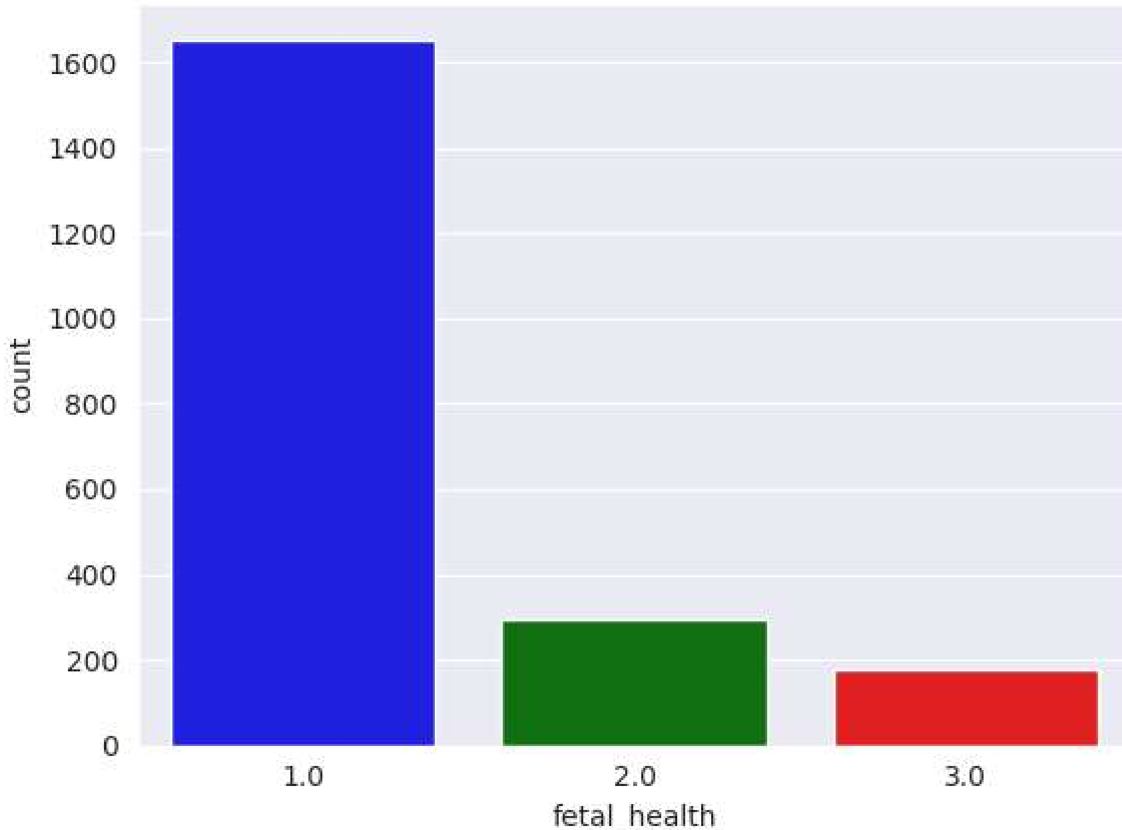
Out[19]:
```
baseline value                                            0
accelerations                                             0
fetal_movement                                            0
uterine_contractions                                      0
light_decelerations                                       0
severe_decelerations                                      0
prolongued_decelerations                                  0
abnormal_short_term_variability                           0
mean_value_of_short_term_variability                      0
percentage_of_time_with_abnormal_long_term_variability    0
mean_value_of_long_term_variability                       0
histogram_width                                           0
histogram_min                                             0
histogram_max                                             0
histogram_number_of_peaks                                 0
histogram_number_of_zeroes                                0
histogram_mode                                            0
histogram_mean                                            0
histogram_median                                          0
histogram_variance                                        0
histogram_tendency                                        0
fetal_health                                              0
dtype: int64
```

In [ ]:
```
#first of all Let us evaluate the target and find out if our data
data['fetal_health'].value_counts()
```

Out[22]:
```
1.0    1655
2.0     295
3.0     176
Name: fetal_health, dtype: int64
```

In [ ]:
```
custom_palette = ['blue', 'green', 'red',]
sns.countplot(data=data, x="fetal_health",palette=custom_palette)
```

Out[33]: <Axes: xlabel='fetal_health', ylabel='count'>



In [ ]:
```
#Milestone 3: Exploratory Data Analysis

#Activity 1: Descriptive statistical analysis
```

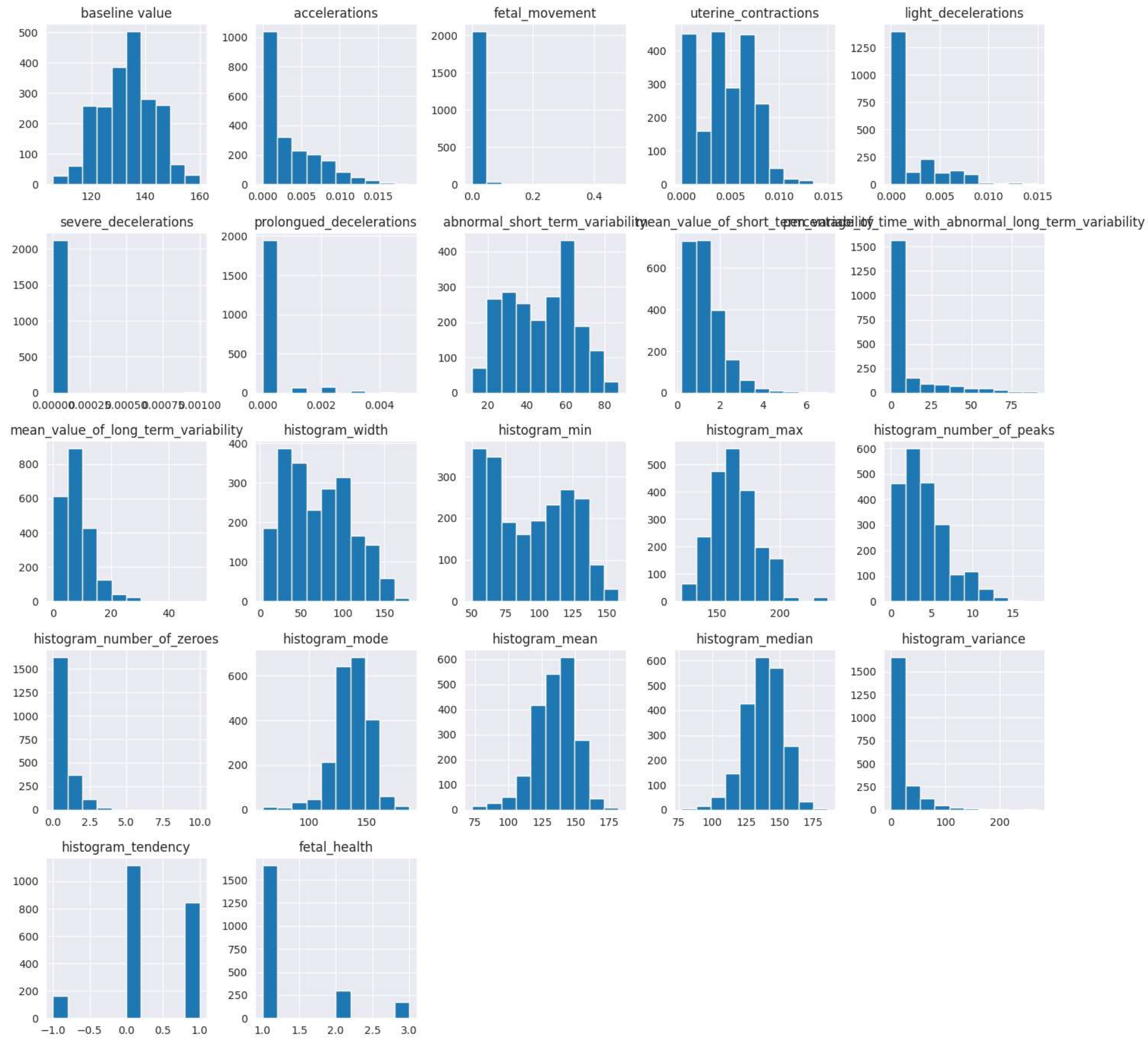
In [ ]:
```
data.nunique()
```

Out[34]:
```
baseline value                                            48
accelerations                                             20
fetal_movement                                           102
uterine_contractions                                      16
light_decelerations                                       16
severe_decelerations                                       2
prolongued_decelerations                                   6
abnormal_short_term_variability                           75
mean_value_of_short_term_variability                      57
percentage_of_time_with_abnormal_long_term_variability    87
mean_value_of_long_term_variability                      249
histogram_width                                          154
histogram_min                                            109
histogram_max                                             86
histogram_number_of_peaks                                 18
histogram_number_of_zeroes                                 9
histogram_mode                                            88
histogram_mean                                           103
histogram_median                                          95
histogram_variance                                       133
histogram_tendency                                         3
fetal_health                                               3
dtype: int64
```

In [ ]:
```

```

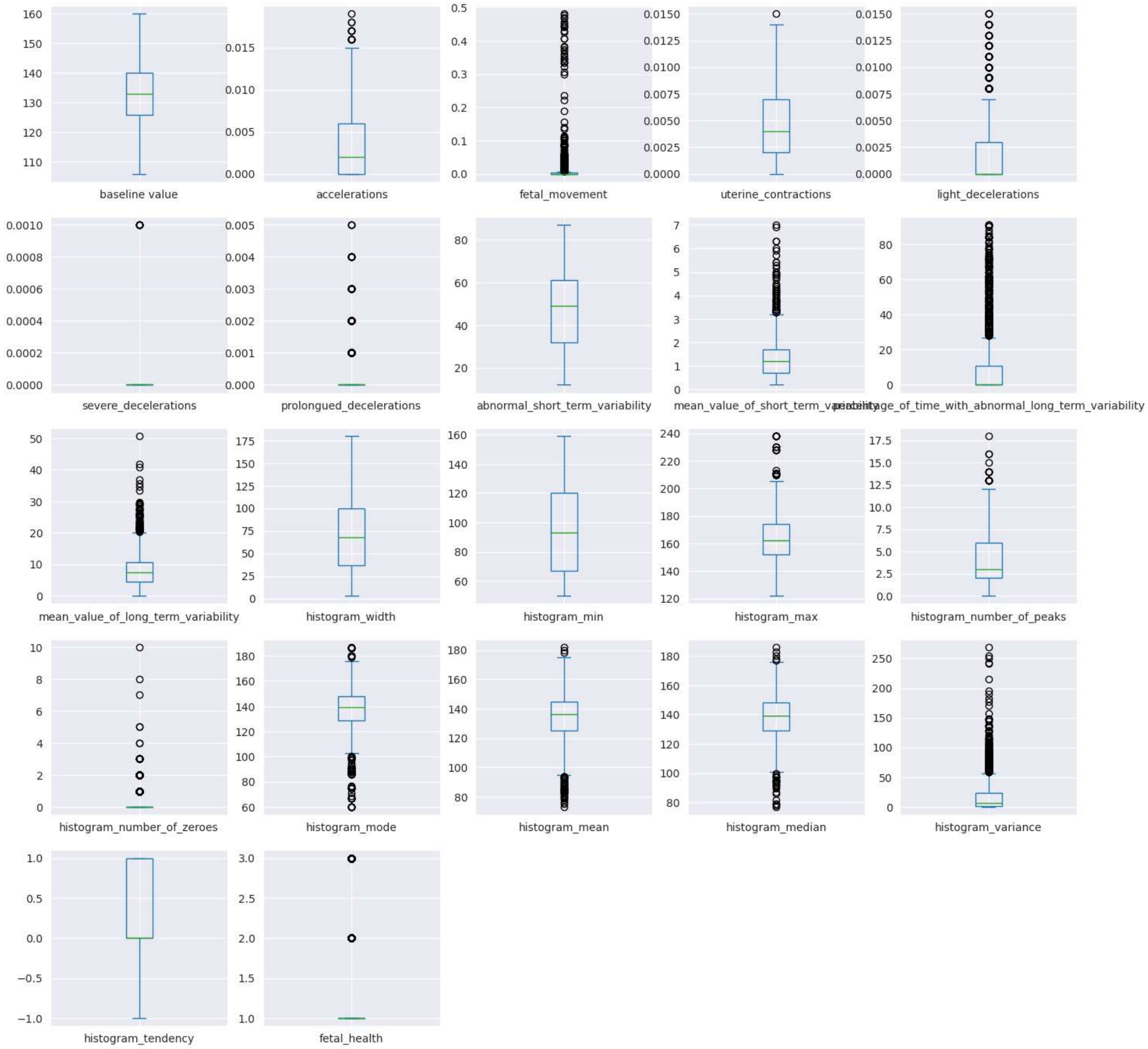
Visual Analysis

In [ ]:
```
data.hist(figsize=(17,17),layout=(5,5),sharex=False);
```



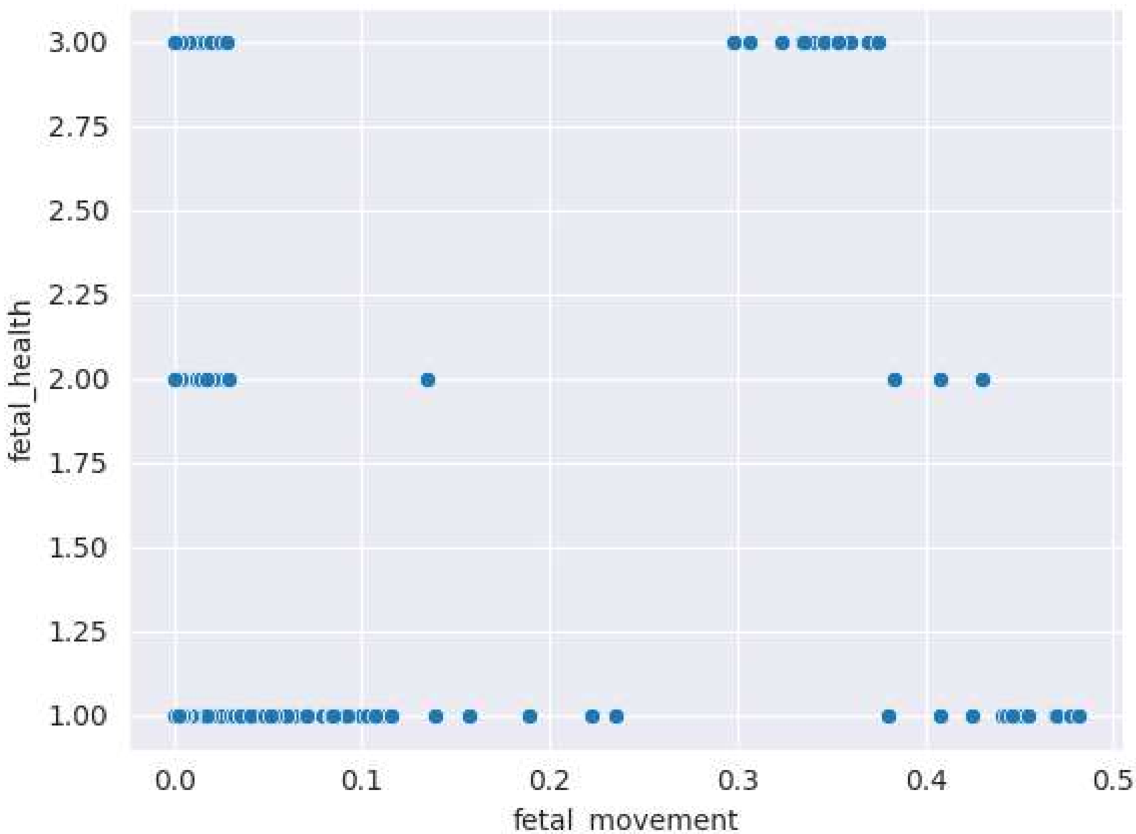
In [ ]:
```
data.plot(kind='box', figsize=(17, 17), layout=(5, 5), sharex=False, subplots=
```

Out[40]:
```
baseline value                                            Axes(0.125,0.747
241;0.133621x0.132759)
accelerations                                           Axes(0.285345,0.747
241;0.133621x0.132759)
fetal_movement                                           Axes(0.44569,0.747
241;0.133621x0.132759)
uterine_contractions                                    Axes(0.606034,0.747
241;0.133621x0.132759)
light_decelerations                                     Axes(0.766379,0.747
241;0.133621x0.132759)
severe_decelerations                                      Axes(0.125,0.587
931;0.133621x0.132759)
prolongued_decelerations                                Axes(0.285345,0.587
931;0.133621x0.132759)
abnormal_short_term_variability                          Axes(0.44569,0.587
931;0.133621x0.132759)
mean_value_of_short_term_variability                    Axes(0.606034,0.587
931;0.133621x0.132759)
percentage_of_time_with_abnormal_long_term_variability    Axes(0.766379,0.587
931;0.133621x0.132759)
mean_value_of_long_term_variability                       Axes(0.125,0.428
621;0.133621x0.132759)
histogram_width                                         Axes(0.285345,0.428
621;0.133621x0.132759)
histogram_min                                            Axes(0.44569,0.428
621;0.133621x0.132759)
histogram_max                                           Axes(0.606034,0.428
621;0.133621x0.132759)
histogram_number_of_peaks                               Axes(0.766379,0.428
621;0.133621x0.132759)
histogram_number_of_zeroes                                 Axes(0.125,0.26
931;0.133621x0.132759)
histogram_mode                                           Axes(0.285345,0.26
931;0.133621x0.132759)
histogram_mean                                            Axes(0.44569,0.26
931;0.133621x0.132759)
histogram_median                                         Axes(0.606034,0.26
931;0.133621x0.132759)
histogram_variance                                       Axes(0.766379,0.26
931;0.133621x0.132759)
histogram_tendency                                           Axes(0.125,
0.11;0.133621x0.132759)
fetal_health                                              Axes(0.285345,
0.11;0.133621x0.132759)
dtype: object
```


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
fetal_health

In [ ]:
```
#bivariate Analysis
import seaborn as sns

# Assuming 'data' is your DataFrame
sns.scatterplot(x=data['fetal_movement'], y=data['fetal_health'])
```

Out[44]: <Axes: xlabel='fetal_movement', ylabel='fetal_health'>



In [ ]:

```
#multivariate analaysis
#correlation matrix
corrmat= data.corr()
plt. figure(figsize=(20,20))
cmap = sns.light_palette("seagreen", as_cmap=True)
sns.heatmap(corrmat,annot=True, cmap=cmap, center=0)
```

Out[51]: <Axes: >

In [ ]:

```
#feature selection
```

In [ ]:
```
data.drop(columns=["histogram_mean"], axis=1, inplace=True)
data.corr()["fetal_health"].sort_values(ascending=False)
```

Out[52]:
```
fetal_health                                              1.000000
prolongued_decelerations                                  0.484859
abnormal_short_term_variability                           0.471191
percentage_of_time_with_abnormal_long_term_variability    0.426146
histogram_variance                                        0.206630
baseline value                                            0.148151
severe_decelerations                                      0.131934
fetal_movement                                            0.088010
histogram_min                                             0.063175
light_decelerations                                       0.058870
histogram_number_of_zeroes                               -0.016682
histogram_number_of_peaks                                -0.023666
histogram_max                                            -0.045265
histogram_width                                          -0.068789
mean_value_of_short_term_variability                     -0.103382
histogram_tendency                                       -0.131976
uterine_contractions                                     -0.204894
histogram_median                                         -0.205033
mean_value_of_long_term_variability                      -0.226797
histogram_mode                                           -0.250412
accelerations                                            -0.364066
Name: fetal_health, dtype: float64
```

In [ ]:
```
columns_to_select = [
    "prolongued_decelerations", "abnormal_short_term_variability",
    "percentage_of_time_with_abnormal_long_term_variability",
    "histogram_variance", "baseline value", "severe_decelerations",
    "fetal_movement", "histogram_min", "light_decelerations",
    "histogram_number_of_zeroes", "histogram_number_of_peaks",
    "histogram_max", "histogram_width", "mean_value_of_short_term_variability"
    "histogram_tendency", "uterine_contractions", "histogram_median",
    "mean_value_of_long_term_variability", "histogram_mode", "accelerations"
]
new_data = data.loc[:, columns_to_select]
```

In [ ]:
```
new_data.head()
```

Out[59]:

	prolongued_decelerations	abnormal_short_term_variability	percentage_of_time_with_abnormal_l
0	0.0	73.0	
1	0.0	17.0	
2	0.0	16.0	
3	0.0	16.0	
4	0.0	16.0	

Scalling the Data

In [66]:
```
from sklearn.preprocessing import MinMaxScaler
X = data.drop(columns=['fetal_health'])
y = data["fetal_health"]
# Instantiating MinMaxScaler
scale = MinMaxScaler()
# Scaling the features in X
X_scaled = pd.DataFrame(scale.fit_transform(X), columns=X.columns)
X_scaled.head()
```

Out[66]:

	baseline value	accelerations	fetal_movement	uterine_contractions	light_decelerations	severe_dece
0	0.259259	0.000000	0.0	0.000000	0.0	
1	0.481481	0.315789	0.0	0.400000	0.2	
2	0.500000	0.157895	0.0	0.533333	0.2	
3	0.518519	0.157895	0.0	0.533333	0.2	
4	0.481481	0.368421	0.0	0.533333	0.0	

In [68]:
```
from sklearn.metrics import accuracy_score, classification_report, confusion_m
from sklearn.model_selection import train_test_split

# Assuming X and y are already defined (features and target variable)

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, rando
print(X_train.shape, X_test.shape)  # Corrected variable names for y_test and 
```

```
(1488, 20) (638, 20)
```

Applying SMOTE for balancing the data

In [69]:
```
!pip install imblearn
```

```
Collecting imblearn
  Downloading imblearn-0.0-py2.py3-none-any.whl (1.9 kB)
Requirement already satisfied: imbalanced-learn in /usr/local/lib/python3.10/
dist-packages (from imblearn) (0.10.1)
Requirement already satisfied: numpy>=1.17.3 in /usr/local/lib/python3.10/dis
t-packages (from imbalanced-learn->imblearn) (1.23.5)
Requirement already satisfied: scipy>=1.3.2 in /usr/local/lib/python3.10/dist
-packages (from imbalanced-learn->imblearn) (1.11.3)
Requirement already satisfied: scikit-learn>=1.0.2 in /usr/local/lib/python3.
10/dist-packages (from imbalanced-learn->imblearn) (1.2.2)
Requirement already satisfied: joblib>=1.1.1 in /usr/local/lib/python3.10/dis
t-packages (from imbalanced-learn->imblearn) (1.3.2)
Requirement already satisfied: threadpoolctl>=2.0.0 in /usr/local/lib/python
3.10/dist-packages (from imbalanced-learn->imblearn) (3.2.0)
Installing collected packages: imblearn
Successfully installed imblearn-0.0
```

In [71]:
```
from imblearn.over_sampling import SMOTE
from collections import Counter
smote = SMOTE()
X_train_smote, y_train_smote = smote.fit_resample(X_train.astype('float'), y_t
print("Before SMOTE:", Counter(y_train))
print("After SMOTE:", Counter(y_train_smote))
```

```
Before SMOTE: Counter({1.0: 1158, 2.0: 201, 3.0: 129})
After SMOTE: Counter({1.0: 1158, 2.0: 1158, 3.0: 1158})
```

Model Building

In [113]:
```
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, confusion_matrix


# Assuming X_train_smote, y_train_smote, and x_test are defined and ready to u

# Create an instance of RandomForestClassifier
RF_model = RandomForestClassifier()

# Fit the model using the training data after SMOTE
RF_model.fit(X_train_smote, y_train_smote)

# Make predictions on the test set
predictions = RF_model.predict(X_test)

# Evaluate the accuracy
RF_accuracy=accuracy_score(y_test, predictions)
print("Accuracy:", accuracy_score(y_test, predictions))

# Create a confusion matrix
pd.crosstab(y_test, predictions)
```

```
Accuracy: 0.9482758620689655
```

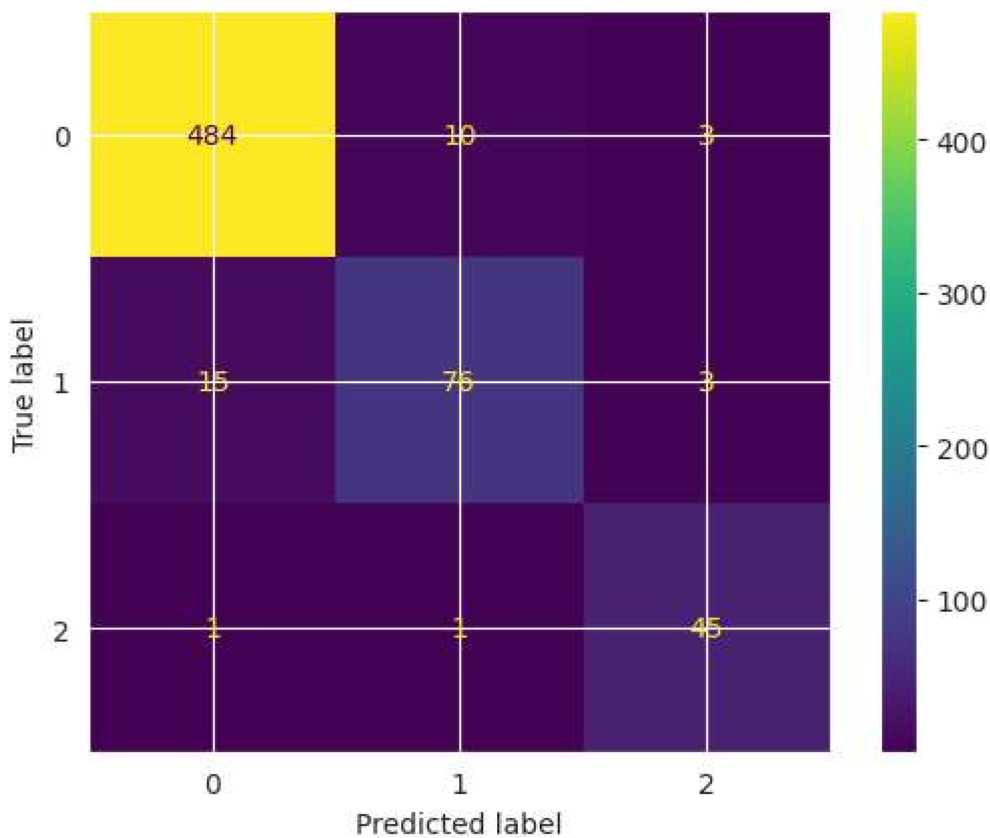
Out[113]:

col_0	1.0	2.0	3.0
fetal_health			
1.0	484	11	2
2.0	15	76	3
3.0	1	1	45

In [84]:

```
from sklearn.metrics import confusion_matrix, ConfusionMatrixDisplay
import matplotlib.pyplot as plt
size = len(X_train_smote)
print("For the amount of training data is:", size)  # Assuming 'size' is defin

print("Accuracy of RandomForestClassifier:", RF_model.score(X_test, y_test))
cm = confusion_matrix(y_test, predictions)

cm_display = ConfusionMatrixDisplay(cm).plot()

plt.show()
```

```
For the amount of training data is: 3474
Accuracy of RandomForestClassifier: 0.9482758620689655
```



In [ ]:

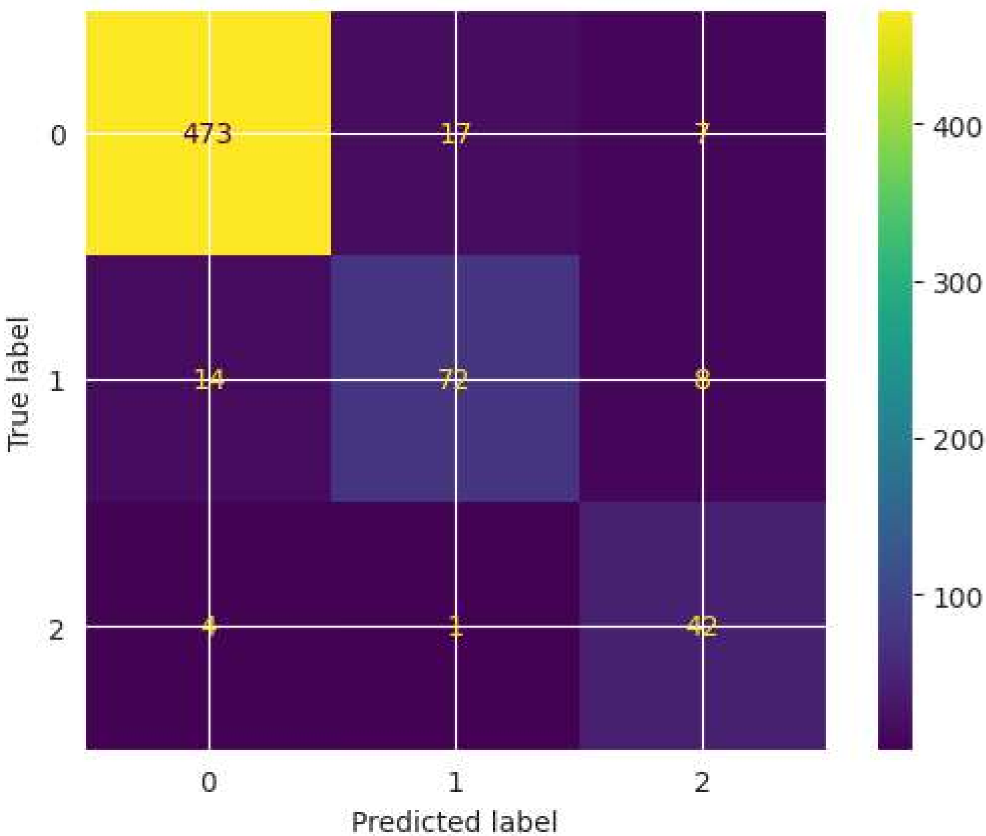
Decision Tree

In [115]:
```
DT_model = DecisionTreeClassifier()
DT_model.fit(X_train_smote, y_train_smote)
predictions = DT_model.predict(X_test)
DT_accuracy=accuracy_score(y_test,predictions)
print(accuracy_score(y_test,predictions))
```

```
0.9247648902821317
```

In [89]:
```

size = len(X_train_smote)
print("For the amount of training data is:", size)  # Assuming 'size' is defin

print("Accuracy of  DecisionTreeClassifier:", DT_model.score(X_test, y_test))
cm = confusion_matrix(y_test, predictions)

cm_display = ConfusionMatrixDisplay(cm).plot()

plt.show()
```

```
For the amount of training data is: 3474
Accuracy of  DecisionTreeClassifier: 0.9200626959247649
```



In [ ]:

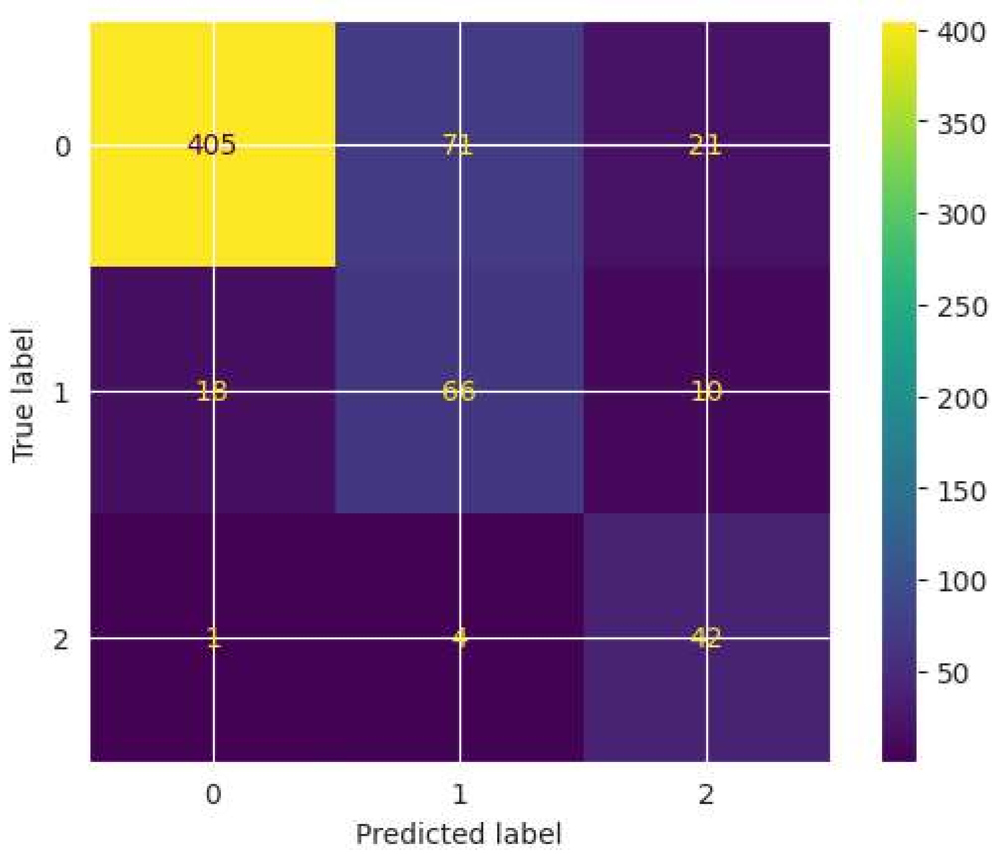
Logistic Regression

In [117]:
```
LR_model = LogisticRegression()
LR_model.fit(X_train_smote, y_train_smote)
predictions = LR_model.predict(X_test)
LR_accuracy=accuracy_score(y_test,predictions)
print(accuracy_score(y_test,predictions))
```

```
0.8040752351097179
```

In [93]:
```
print("For the amounts of training data is: ",size)

print("Accuracy of LogisticRegression:",LR_model.score(X_test,y_test))
cm = confusion_matrix(y_test, predictions)

cm_display = ConfusionMatrixDisplay(cm).plot()

plt.show()
```

```
For the amounts of training data is:  3474
Accuracy of LogisticRegression: 0.8040752351097179
```

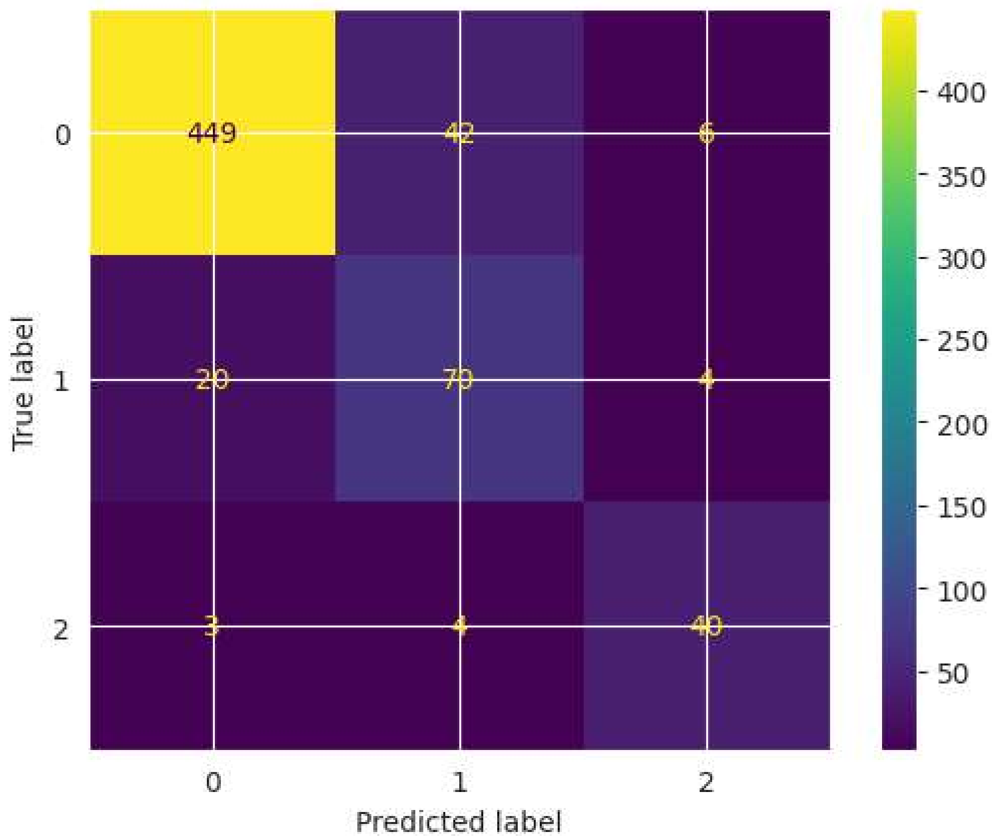


In [118]:
```
#KNeighborsClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import accuracy_score

# Assuming X_train_smote, y_train_smote, and X_test are defined and ready to u

# Create an instance of KNeighborsClassifier with 5 neighbors
KNN_model = KNeighborsClassifier(n_neighbors=5)

# Fit the KNN model using the training data after SMOTE
KNN_model.fit(X_train_smote, y_train_smote)

# Make predictions on the test set
predictions = KNN_model.predict(X_test)

# Calculate and print the accuracy
KN_accuracy=accuracy_score(y_test, predictions)
print("Accuracy:", accuracy_score(y_test, predictions))
```

```
Accuracy: 0.8761755485893417
```

In [97]:
```
print("For the amounts of training data is: ",size)

print("Accuracy of KNeighborsClassifier: ",KNN_model.score(X_test,y_test))
cm = confusion_matrix(y_test, predictions)

cm_display = ConfusionMatrixDisplay(cm).plot()

plt.show()
```

```
For the amounts of training data is:  3474
Accuracy of KNeighborsClassifier:  0.8761755485893417
```



In [119]:
```
#performance Testing
names = ['RandomForestClassifier', 'KNeighborsClassifier',"LogisticRegression"
scores = [RF_accuracy, KN_accuracy,LR_accuracy,DT_accuracy]

# Create a DataFrame to display names and scores
df = pd.DataFrame()
df['name'] = names
df['score'] = scores
df
```

Out[119]:

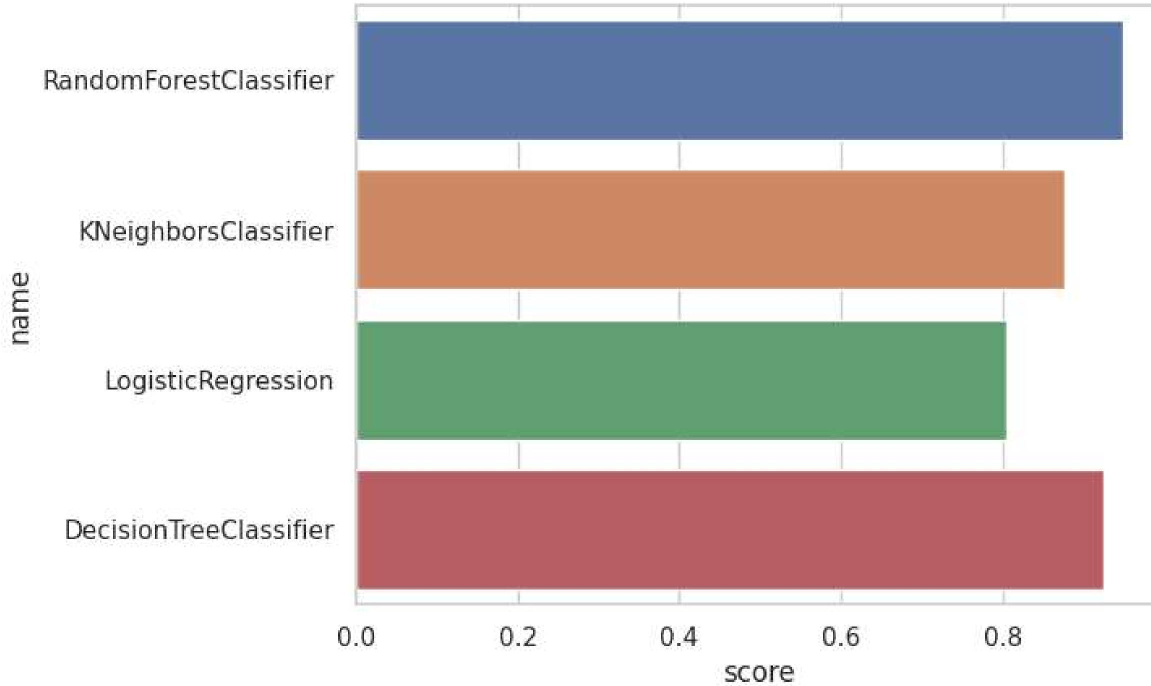
	name	score
0	RandomForestClassifier	0.948276
1	KNeighborsClassifier	0.876176
2	LogisticRegression	0.804075
3	DecisionTreeClassifier	0.924765

In [120]:
```
CM=sns.light_palette("red",as_cmap=True)
C=df.style.background_gradient(cmap=CM)
C
```

Out[120]:

	name	score
0	RandomForestClassifier	0.948276
1	KNeighborsClassifier	0.876176
2	LogisticRegression	0.804075
3	DecisionTreeClassifier	0.924765

In [121]:
```
sns.set(style="whitegrid")
ax=sns.barplot(y="name", x="score", data=df)
```



In [123]:
```
# saving the model

import pickle
pickle.dump(RF_model,open('fetal_health1.pkl1','wb'))
```

In [ ]:
```
```

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

model=pickle.load(open(r'fetal_health1.pkl','rb'))
app-Flask (name)


@app.route("/")
def f():
return render_template("index.html")
@app.route("/home", methods=["GET", "POST"])

def home():
f
prolongued decelerations float(request.form['prolongued decelerations'])

abnormal_short_term_variability float (request.form['abnormal_short_term
variability']) percentage_of_time_with_abnormal_long term_variability
float(request.form['percentage of time'])

histogram variance float(request.form['histogram variance']) histogram median
float(request.fowl'histogram_median'])

mean_value_of_long_term_variability
float(request.form['mean_value_of_long_term_variability'])

histogram mode
float(request.form['histogram_mode']) accelerations
float(request.form['accelerations'])

x= [[prolongued_decelerations, abnormal_short_term_variability,
percentage_of_time_with abnormal]]

output=model.predict(x)

out=['Normal','Pathological','Suspect']

if int (output[0])--0:

output='Normal elif int (output[0]) 1:

output=Pathological

else: output='Suspect

return render_template('output.html',output=output)
if name "main : 21 app.run(debug=True)
```