## TABLE OF CONTENTS

**CHAPTER TITLE PAGE**

**NO. NO.**

1. **INTRODUCTION 4**
2. **OBJECTIVES** **5**
3. **DATASET** **6**
4. **METHODOLOGY 9**
5. **CODE AND RESULT** **10**
6. **RESULTS 28**

# CHAPTER 1

## INTRODUCTION

Fetal health refers to the well-being of a developing fetus during pregnancy. It encompasses a variety of factors including fetal growth and development, fetal organ function, and fetal response to stressors.

Foetal movement, foetal heart rate, and amniotic fluid content are three crucial aspects to consider while evaluating the health of the foetus. Foetal movement is a sign of the health and neurologic function of the developing foetus. Foetal distress or compromise may be indicated by a decrease in foetal movement. Another crucial sign of foetal health is the heart rate, with fluctuations in the heart rate revealing the level of stress experienced by the foetus. The amount of amniotic fluid is also measured since low amounts of amniotic fluid may be a sign of foetal impairment.

Throughout pregnancy, foetal health can be monitored using a variety of medical tests and procedures. Medical experts can track foetal growth and development by using ultrasound exams to provide fine-grained images of the growing foetus. Additionally, non-invasive techniques like the non-stress test or more invasive techniques like foetal scalp electrode monitoring can be used to monitor the foetal heart rate.

Healthy foetal development is crucial for a healthy pregnancy and delivery. Regular check-ups with medical specialists as part of prenatal care can help identify potential issues early on and enable rapid medical action when necessary. This can increase the likelihood of a healthy pregnancy and birth as well as help assure the new-born’s long-term wellbeing.

There are several machine learning algorithms and models that can be used to predict **fetal health**, including ***decision trees, support vector machines (SVM), neural networks, and random forests***. These algorithms use various features and input data to make predictions about fetal health and can assist in identifying potential risks or complications during pregnancy.

# CHAPTER 2

## OBJECTIVE

The objectives of fetal health are primarily centered around ensuring the well-being of the developing fetus during pregnancy. These objectives can be broken down into three main categories: identifying potential risks to fetal health, managing identified risks, and promoting overall fetal development and health.

Identifying potential risks to fetal health involves monitoring the mother and fetus for any signs of potential complications. This may involve regular check-ups with healthcare professionals, as well as various medical tests and procedures to assess fetal health. These tests can help identify potential risks such as fetal distress, intrauterine growth restriction, and preterm labor, which can be managed to ensure the best possible outcome for mother and baby.

Managing identified risks involves taking appropriate steps to mitigate any risks that have been identified. This may involve medical intervention, such as medication or surgery, or lifestyle changes such as reducing stress or improving nutrition. Proper management of identified risks can help reduce the risk of complications during pregnancy and improve the overall health of the developing fetus.

Promoting overall fetal development and health involves ensuring that the fetus is receiving adequate nutrition and is developing normally. This may involve encouraging healthy lifestyle choices in the mother, such as maintaining a healthy diet and exercise regimen, as well as monitoring fetal growth and development through regular medical check-ups. Promoting overall fetal health can help reduce the risk of complications during pregnancy and can improve the long-term health outcomes for the developing fetus.

# CHAPTER 3

**DATASET**

Reduction of child mortality is reflected in several of the United Nations' Sustainable Development Goals and is a key indicator of human progress.

The UN expects that by 2030, countries end preventable deaths of newborns and children under 5 years of age, with all countries aiming to reduce under-5 mortality to at least as low as 25 per 1,000 live births.

Parallel to notion of child mortality is of course maternal mortality, which accounts for **295 000 deaths** during and following pregnancy and childbirth (as of 2017). The vast majority of these deaths **(94%)** occurred in low-resource settings, and most **could have been prevented**. In light of what was mentioned above, **Cardiotocograms (CTGs)** are a simple and cost accessible option to assess fetal health, allowing healthcare professionals to take action in order to prevent child and maternal mortality. The equipment itself works by sending ultrasound pulses and reading its response, thus shedding light on fetal heart rate (FHR), fetal movements, uterine contractions and more.

This dataset contains **2126** records of features extracted from Cardiotocogram exams, which were then classified by three expert obstetritians into **3 classes**:

* Normal
* Suspect
* Pathological

# Column Non-Null Count Dtype

--- ------ -------------- -----

1. accelerations 2112 non-null float64
2. fetal\_movement 2112 non-null float64
3. uterine\_contractions 2112 non-null float64
4. light\_decelerations 2112 non-null float64
5. severe\_decelerations 2112 non-null float64
6. prolongued\_decelerations 2112 non-null float64
7. abnormal\_short\_term\_variability 2112 non-null float64
8. mean\_value\_of\_short\_term\_variability 2112 non-null float64
9. percentage\_of\_time\_with\_abnormal\_long\_term\_variability 2112 non-null float64
10. mean\_value\_of\_long\_term\_variability 2112 non-null float64
11. histogram\_width 2112 non-null float64
12. histogram\_min 2112 non-null float64
13. histogram\_max 2112 non-null float64
14. histogram\_number\_of\_peaks 2112 non-null float64
15. histogram\_number\_of\_zeroes 2112 non-null float64
16. histogram\_mode 2112 non-null float64
17. histogram\_mean 2112 non-null float64
18. histogram\_median 2112 non-null float64
19. histogram\_variance 2112 non-null float64
20. histogram\_tendency 2112 non-null float64
21. fetal\_health 2112 non-null

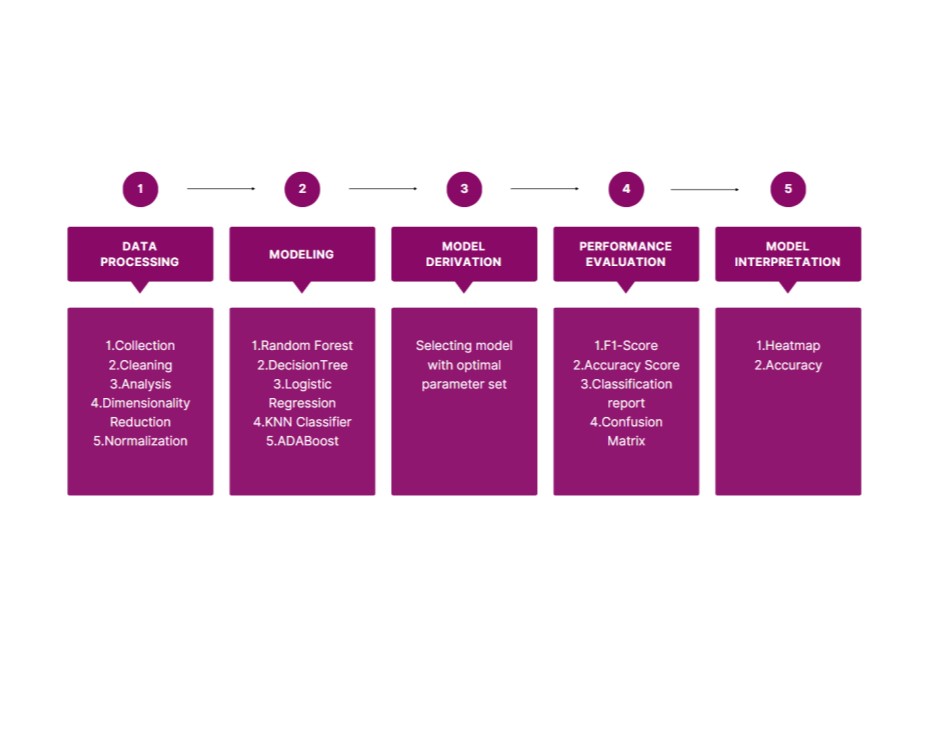
The Fetal Health Classification dataset is a medical dataset that contains features extracted from Cardiotocogram (CTG) measurements, which are used to monitor the fetal heart rate (FHR) and uterine contractions during pregnancy. The dataset contains 2,126 records of fetal health and consists of 21 features, which are described below:

1. baseline value: The FHR baseline value in beats per minute (bpm)
2. accelerations: The number of accelerations per second
3. fetal\_movement: The number of fetal movements per second
4. uterine\_contractions: The number of uterine contractions per second
5. light\_decelerations: The number of light decelerations per second
6. severe\_decelerations: The number of severe decelerations per second
7. prolongued\_decelerations: The number of prolonged decelerations per second
8. abnormal\_short\_term\_variability: The percentage of time with abnormal short-term variability
9. mean\_value\_of\_short\_term\_variability: The mean value of short-term variability
10. percentage\_of\_time\_with\_abnormal\_long\_term\_variability: The percentage of time with abnormal long-term variability
11. mean\_value\_of\_long\_term\_variability: The mean value of long-term variability
12. histogram\_width: The width of FHR histogram
13. histogram\_min: The minimum value of FHR histogram
14. histogram\_max: The maximum value of FHR histogram
15. histogram\_number\_of\_peaks: The number of histogram peaks
16. histogram\_number\_of\_zeroes: The number of histogram zeroes
17. histogram\_mode: The histogram mode value
18. histogram\_mean: The histogram mean value
19. histogram\_median: The histogram median value
20. histogram\_variance: The histogram variance value
21. histogram\_tendency: The histogram tendency value

The dataset is labeled with three classes, namely, Normal (N), Suspect (S), and Pathological (P), which represent the fetal health status. The aim is to use the features to predict the fetal health status accurately. This dataset can be useful in developing decision support systems for fetal health monitoring during pregnancy

# CHAPTER 4

## METHODOLOGY



# CHAPTER 5

## CODE AND RESULT

*# This Python 3 environment comes with many helpful analytics libraries installed*

*# It is defined by the kaggle/python Docker image: https://github.com/kaggle/docker-python*

*# For example, here's several helpful packages to load*

**import** numpy **as** np *# linear algebra*

**import** pandas **as** pd *# data processing, CSV file I/O (e.g. pd.read\_csv)*

*# Input data files are available in the read-only "../input/" directory* *# For example, running this (by clicking run or pressing Shift+Enter) will list all files under the input directory*

**import** os **for** dirname, \_, filenames **in** os.walk('/kaggle/input'): **for** filename **in** filenames:

print(os.path.join(dirname, filename))

*# You can write up to 20GB to the current directory (/kaggle/working/) that gets preserved as output when you create a version using "Save & Run All"*  *# You can also write temporary files to /kaggle/temp/, but they won't be saved outside of the current session*

/kaggle/input/fetal-health-classification/fetal\_health.csv

**import** pandas **as** pd **import** numpy **as** np **import** matplotlib.pyplot **as** plt **import** seaborn **as** sns **from** sklearn.preprocessing **import** StandardScaler **from** sklearn.model\_selection **import** train\_test\_split **from** sklearn.ensemble **import** RandomForestClassifier **from** sklearn.metrics **import** confusion\_matrix, accuracy\_score **from** tensorflow.keras.callbacks **import** EarlyStopping early\_stop = EarlyStopping(monitor='val\_loss', patience=10)

Loading the dataset

df = pd.read\_csv('/kaggle/input/fetal-healthclassification/fetal\_health.csv') df=df.iloc[:,1:]

df=df.drop\_duplicates() df.head()

accelerations fetal\_movement uterine\_contractions light\_decelerations

\

1. 0.000 0.0 0.000 0.000
2. 0.006 0.0 0.006 0.003
3. 0.003 0.0 0.008 0.003
4. 0.003 0.0 0.008 0.003 4 0.007 0.0 0.008 0.000

severe\_decelerations prolongued\_decelerations \ 0 0.0 0.0

1. 0.0 0.0
2. 0.0 0.0
3. 0.0 0.0
4. 0.0 0.0

abnormal\_short\_term\_variability mean\_value\_of\_short\_term\_variability \ 0 73.0 0.5

1. 17.0 2.1
2. 16.0 2.1
3. 16.0 2.4 4 16.0 2.4

percentage\_of\_time\_with\_abnormal\_long\_term\_variability \ 0 43.0

1. 0.0
2. 0.0
3. 0.0 4 0.0

mean\_value\_of\_long\_term\_variability ... histogram\_min histogram\_max \ 0 2.4 ... 62.0 126.0

1. 10.4 ... 68.0 198.0
2. 13.4 ... 68.0 198.0
3. 23.0 ... 53.0 170.0 4 19.9 ... 53.0 170.0

histogram\_number\_of\_peaks histogram\_number\_of\_zeroes histogram\_mode \ 0 2.0 0.0 120.0

1. 6.0 1.0 141.0
2. 5.0 1.0 141.0
3. 11.0 0.0 137.0 4 9.0 0.0 137.0

histogram\_mean histogram\_median histogram\_variance histogram\_tendency

\

1. 137.0 121.0 73.0 1.0
2. 136.0 140.0 12.0 0.0
3. 135.0 138.0 13.0 0.0
4. 134.0 137.0 13.0 1.0 4 136.0 138.0 11.0 1.0

fetal\_health 0 2.0

1. 1.0
2. 1.0
3. 1.0
4. 1.0

[5 rows x 21 columns]

Exploratory Data Analysis df.shape (2112, 21) df.info()

<class 'pandas.core.frame.DataFrame'> Int64Index: 2112 entries, 0 to 2125 Data columns (total 21 columns):

# Column Non-Null Count Dtype

--- ------ -------------- -----

1. accelerations 2112 non-null float64
2. fetal\_movement 2112 non-null float64
3. uterine\_contractions 2112 non-null float64
4. light\_decelerations 2112 non-null float64
5. severe\_decelerations 2112 non-null float64
6. prolongued\_decelerations 2112 non-null float64
7. abnormal\_short\_term\_variability 2112 non-null float64
8. mean\_value\_of\_short\_term\_variability 2112 non-null float64
9. percentage\_of\_time\_with\_abnormal\_long\_term\_variability 2112 non-null float64
10. mean\_value\_of\_long\_term\_variability 2112 non-null float64
11. histogram\_width 2112 non-null float64
12. histogram\_min 2112 non-null float64
13. histogram\_max 2112 non-null float64
14. histogram\_number\_of\_peaks 2112 non-null float64
15. histogram\_number\_of\_zeroes 2112 non-null float64
16. histogram\_mode 2112 non-null float64
17. histogram\_mean 2112 non-null float64
18. histogram\_median 2112 non-null float64
19. histogram\_variance 2112 non-null float64
20. histogram\_tendency 2112 non-null float64
21. fetal\_health 2112 non-null int64 dtypes: float64(20), int64(1) memory usage: 363.0 KB df.isnull().sum()

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 df.describe()

accelerations fetal\_movement uterine\_contractions \ count 2112.000000 2112.000000 2112.000000 mean 0.003190 0.009511 0.004389 std 0.003872 0.046814 0.002940 min 0.000000 0.000000 0.000000 25% 0.000000 0.000000 0.002000

50% 0.002000 0.000000 0.005000 75% 0.006000 0.003000 0.007000 max 0.019000 0.481000 0.015000

light\_decelerations severe\_decelerations prolongued\_decelerations \ count 2112.000000 2112.000000 2112.000000 mean 0.001902 0.000003 0.000160 std 0.002966 0.000057 0.000592 min 0.000000 0.000000 0.000000 25% 0.000000 0.000000 0.000000

50% 0.000000 0.000000 0.000000 75% 0.003000 0.000000 0.000000 max 0.015000 0.001000 0.005000

abnormal\_short\_term\_variability mean\_value\_of\_short\_term\_variability

\

count 2112.000000 2112.000000 mean 46.981061 1.335511 std 17.171788 0.884290 min 12.000000 0.200000 25% 32.000000 0.700000

50% 49.000000 1.200000 75% 61.000000 1.700000 max 87.000000 7.000000

percentage\_of\_time\_with\_abnormal\_long\_term\_variability \ count 2112.000000 mean 9.773201 std 18.313812 min 0.000000 25% 0.000000 50% 0.000000 75% 11.000000 max 91.000000

mean\_value\_of\_long\_term\_variability ... histogram\_min histogram\_max

\

count 2112.000000 ... 2112.000000 2112.000000 mean 8.167472 ... 93.546875 164.103693 std 5.634115 ... 29.558037 17.948559 min 0.000000 ... 50.000000 122.000000 25% 4.600000 ... 67.000000 152.000000

50% 7.400000 ... 93.000000 162.000000 75% 10.800000 ... 120.000000 174.000000 max 50.700000 ... 159.000000 238.000000

histogram\_number\_of\_peaks histogram\_number\_of\_zeroes histogram\_mode

\

count 2112.000000 2112.000000 2112.000000 mean 4.077178 0.325758 137.448390 std 2.952363 0.707903 16.403636 min 0.000000 0.000000 60.000000 25% 2.000000 0.000000 129.000000

50% 4.000000 0.000000 139.000000 75% 6.000000 0.000000 148.000000 max 18.000000 10.000000 187.000000

histogram\_mean histogram\_median histogram\_variance \ count 2112.000000 2112.000000 2112.000000 mean 134.592330 138.083333 18.916193 std 15.610519 14.479658 29.042726 min 73.000000 77.000000 0.000000 25% 125.000000 129.000000 2.000000

50% 136.000000 139.000000 7.000000 75% 145.000000 148.000000 24.000000 max 182.000000 186.000000 269.000000

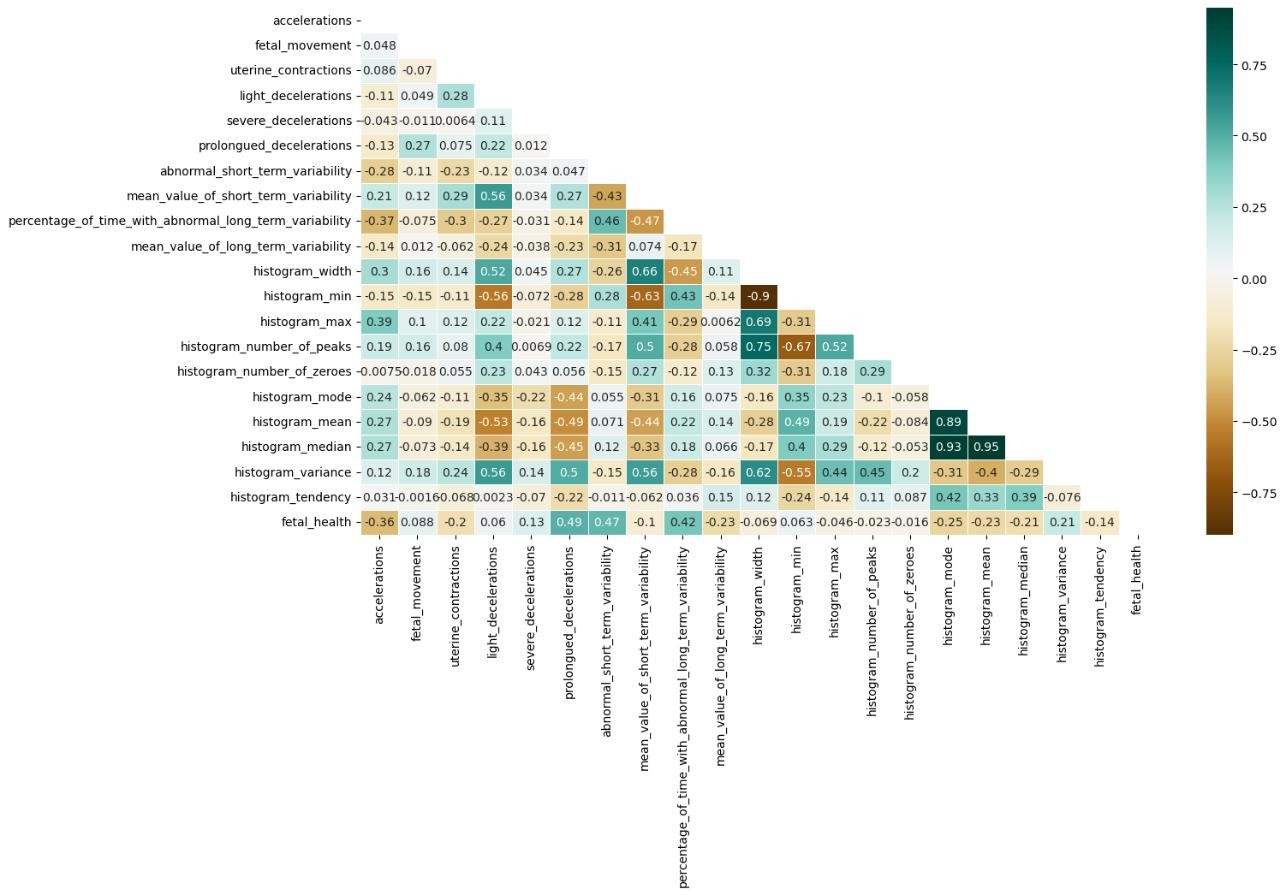
histogram\_tendency fetal\_health count 2112.000000 2112.000000 mean 0.318182 1.303504 std 0.611039 0.614237 min -1.000000 1.000000 25% 0.000000 1.000000

50% 0.000000 1.000000 75% 1.000000 1.000000 max 1.000000 3.000000

[8 rows x 21 columns]

Correlation plot corr = df.corr() plt.figure(figsize=(15,8))

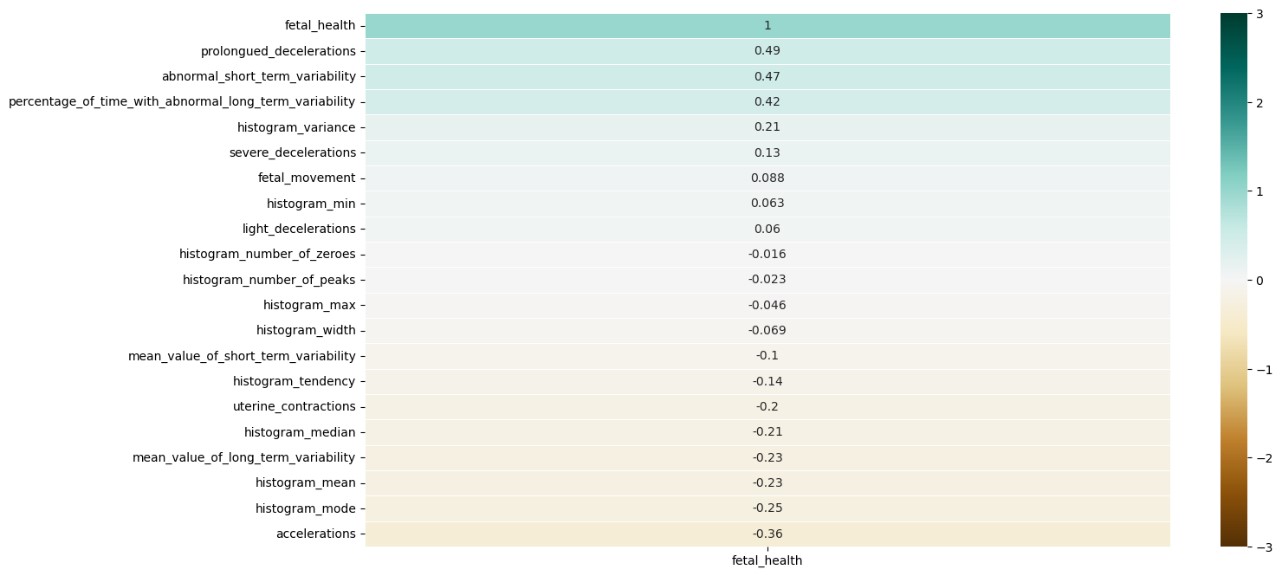
mask = np.triu(np.ones\_like(corr, dtype=bool)) sns.heatmap(corr,mask=mask,cmap='BrBG',annot=True,linewidth=.5,square=False) <AxesSubplot:>



Correlation of all columns with respect to class plt.figure(figsize=(15,8))

sns.heatmap(df.corr()[['fetal\_health']].sort\_values(by='fetal\_health', ascending=False),cmap='BrBG', vmin=-3, vmax= 3 , center=0, annot=True,linewidth=.5,square=False)

<AxesSubplot:>



x=df.drop('fetal\_health',axis=1) y=df['fetal\_health']

**from** sklearn.preprocessing **import** LabelEncoder le=LabelEncoder() df['fetal\_health']=le.fit\_transform(df['fetal\_health'])

**for** column **in** x.columns: x[column] = (x[column] - x[column].min()) / (x[column].max() - x[column].min()) x.head()

accelerations fetal\_movement uterine\_contractions light\_decelerations

\

1. 0.000000 0.0 0.000000 0.0
2. 0.315789 0.0 0.400000 0.2
3. 0.157895 0.0 0.533333 0.2
4. 0.157895 0.0 0.533333 0.2 4 0.368421 0.0 0.533333 0.0

severe\_decelerations prolongued\_decelerations \ 0 0.0 0.0

1. 0.0 0.0
2. 0.0 0.0
3. 0.0 0.0
4. 0.0 0.0

abnormal\_short\_term\_variability mean\_value\_of\_short\_term\_variability \ 0 0.813333 0.044118

1. 0.066667 0.279412
2. 0.053333 0.279412
3. 0.053333 0.323529 4 0.053333 0.323529

percentage\_of\_time\_with\_abnormal\_long\_term\_variability \

1. 0.472527
2. 0.000000
3. 0.000000
4. 0.000000
5. 0.000000

mean\_value\_of\_long\_term\_variability histogram\_width histogram\_min \ 0 0.047337 0.344633 0.110092

1. 0.205128 0.717514 0.165138
2. 0.264300 0.717514 0.165138
3. 0.453649 0.644068 0.027523 4 0.392505 0.644068 0.027523

histogram\_max histogram\_number\_of\_peaks histogram\_number\_of\_zeroes \ 0 0.034483 0.111111 0.0

1. 0.655172 0.333333 0.1
2. 0.655172 0.277778 0.1
3. 0.413793 0.611111 0.0 4 0.413793 0.500000 0.0

histogram\_mode histogram\_mean histogram\_median histogram\_variance \ 0 0.472441 0.587156 0.403670 0.271375

1. 0.637795 0.577982 0.577982 0.044610
2. 0.637795 0.568807 0.559633 0.048327
3. 0.606299 0.559633 0.550459 0.048327 4 0.606299 0.577982 0.559633 0.040892

histogram\_tendency 0 1.0

1. 0.5
2. 0.5
3. 1.0 4 1.0

**from** sklearn.model\_selection **import** train\_test\_split

xtrain,xtest,ytrain,ytest= train\_test\_split(x,y,test\_size=0.1,stratify=y) print(xtrain.shape) print(xtest.shape) print(ytrain.shape) print(ytest.shape)

(1900, 20)

(212, 20) (1900,) (212,)

**from** sklearn.decomposition **import** PCA

pca = PCA()

x\_train = pca.fit\_transform(X\_train) x\_test = pca.transform(X\_test)

explained\_variance = pca.explained\_variance\_ratio\_ explained\_variance

array([5.98047433e-01, 1.57319290e-01, 9.60723218e-02, 7.17305287e-02, 3.61564735e-02, 2.83195995e-02, 5.99169535e-03, 3.95019578e-03, 1.47454710e-03, 7.61650212e-04, 9.17104053e-05, 5.74755426e-05,

2.66198715e-05, 4.55371086e-07, 1.71858342e-09, 1.40774917e-09,

5.79359987e-10, 3.70914199e-11, 7.46480149e-13, 3.24768840e-32])

Decision Tree **from** sklearn.tree **import** DecisionTreeClassifier

dtc = DecisionTreeClassifier(random\_state=42, max\_depth=7) dtc = dtc.fit(X\_train, y\_train) y\_pred\_dtc = dtc.predict(X\_test)

**from** sklearn.metrics **import** \*

dtc\_acc = accuracy\_score(y\_test, y\_pred\_dtc) dtc\_acc

0.9290780141843972 print(classification\_report(y\_test, y\_pred\_dtc))

precision recall f1-score support

1.0 0.94 0.97 0.96 330 2.0 0.79 0.72 0.76 58

3.0 1.00 0.89 0.94 35

accuracy 0.93 423 macro avg 0.91 0.86 0.88 423 weighted avg 0.93 0.93 0.93 423

print(confusion\_matrix(y\_test, y\_pred\_dtc))

[[320 10 0]

[ 16 42 0] [ 3 1 31]]

f1\_micro = f1\_score(y\_test, y\_pred\_dtc, average='micro') print("F1-Score: ", f1\_micro) F1-Score: 0.9290780141843973

f1\_macro = f1\_score(y\_test, y\_pred\_dtc, average='macro') print("F1-Score: ", f1\_macro)

F1-Score: 0.8842674717114178

f1\_weighted = f1\_score(y\_test, y\_pred\_dtc,average='weighted') print("F1-Score: ", f1\_weighted) F1-Score: 0.9278150048114746 Random Forest Classifier

**from** sklearn.ensemble **import** RandomForestClassifier rfc = RandomForestClassifier(n\_estimators=100, random\_state=42) rfc = rfc.fit(X\_train, y\_train) y\_pred\_rfc = rfc.predict(X\_test) f1\_rfc = f1\_score(y\_test, y\_pred\_rfc,average='weighted')

**from** sklearn.ensemble **import** RandomForestClassifier rfc = RandomForestClassifier(n\_estimators=100, random\_state=42) rfc = rfc.fit(X\_train, y\_train) y\_pred\_rfc = rfc.predict(X\_test) f1\_micro = f1\_score(y\_test, y\_pred\_rfc,average='micro')

**from** sklearn.ensemble **import** RandomForestClassifier rfc = RandomForestClassifier(n\_estimators=100, random\_state=42) rfc = rfc.fit(X\_train, y\_train) y\_pred\_rfc = rfc.predict(X\_test) f1\_macro = f1\_score(y\_test, y\_pred\_rfc,average='macro')

print("F1-Score: ", f1\_weighted) print("Accuracy: ", accuracy\_score(y\_test, y\_pred\_rfc))

F1-Score: 0.9278150048114746

Accuracy: 0.9527186761229315

print("F1-Score: ", f1\_micro) print("Accuracy: ", accuracy\_score(y\_test, y\_pred\_rfc))

F1-Score: 0.9527186761229315

Accuracy: 0.9527186761229315

print("F1-Score: ", f1\_macro) print("Accuracy: ", accuracy\_score(y\_test, y\_pred\_rfc))

F1-Score: 0.9116546866393908 Accuracy: 0.9527186761229315 print(classification\_report(y\_test, y\_pred\_rfc))

precision recall f1-score support

1.0 0.96 0.99 0.97 330 2.0 0.91 0.74 0.82 58

3.0 0.97 0.91 0.94 35

accuracy 0.95 423 macro avg 0.95 0.88 0.91 423 weighted avg 0.95 0.95 0.95 423

print(confusion\_matrix(y\_test, y\_pred\_rfc))

[[328 2 0]

[ 14 43 1]

[ 1 2 32]]

### LOGISTIC REGRESSION

**from** sklearn.linear\_model **import** LogisticRegression size = X\_train.shape[0]

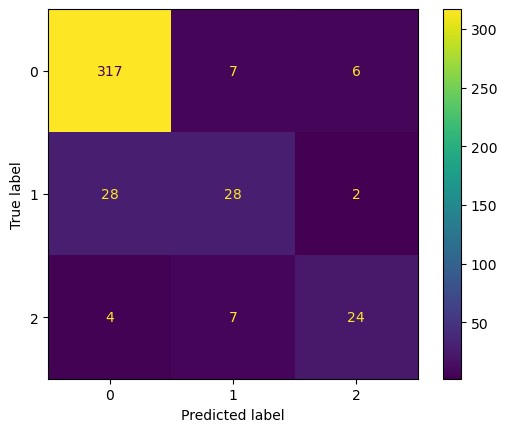
model = LogisticRegression(max\_iter=1000, C=0.009, penalty="l2", solver="newton-cg") model.fit(X\_train, y\_train)

print("For the amounts of training data is: ",size)

print("Accuracy of LogisticRegression: ",model.score(X\_test,y\_test)) y\_pred = model.predict(X\_test) cm = confusion\_matrix(y\_test, y\_pred) cm\_display = ConfusionMatrixDisplay(cm).plot() plt.show()

For the amounts of training data is: 1689

Accuracy of LogisticRegression: 0.8723404255319149



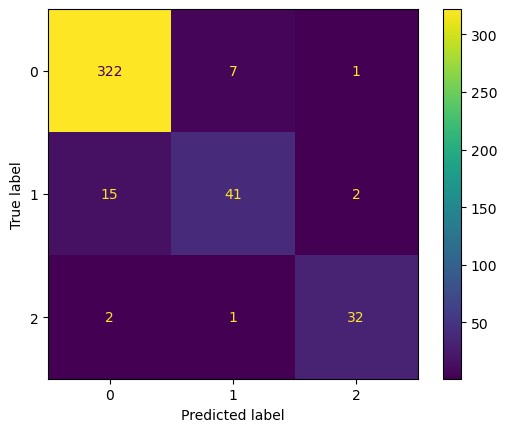
### DECISION TREE CLASSIFIER

model = DecisionTreeClassifier() model.fit(X\_train, y\_train)

print("For the amounts of training data is: ",size) print("Accuracy of DecisionTree: ",model.score(X\_test, y\_test)) y\_pred = model.predict(X\_test) cm = confusion\_matrix(y\_test, y\_pred) cm\_display = ConfusionMatrixDisplay(cm).plot() plt.show()

For the amounts of training data is: 1689

Accuracy of DecisionTree: 0.933806146572104

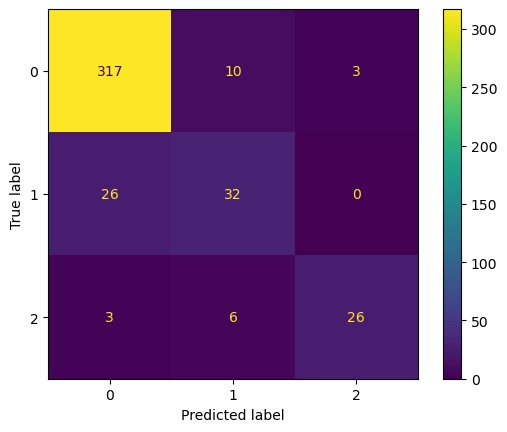


### KNNEIGHBORS CLASSIFIER

**from** sklearn.neighbors **import** KNeighborsClassifier model = KNeighborsClassifier(n\_neighbors=5) model.fit(X\_train, y\_train)

print("For the amounts of training data is: ",size) print("Accuracy of K-NN:",model.score(X\_test, y\_test)) y\_pred = model.predict(X\_test) cm = confusion\_matrix(y\_test, y\_pred) cm\_display = ConfusionMatrixDisplay(cm).plot() plt.show()

For the amounts of training data is: 1689 Accuracy of K-NN: 0.8865248226950354



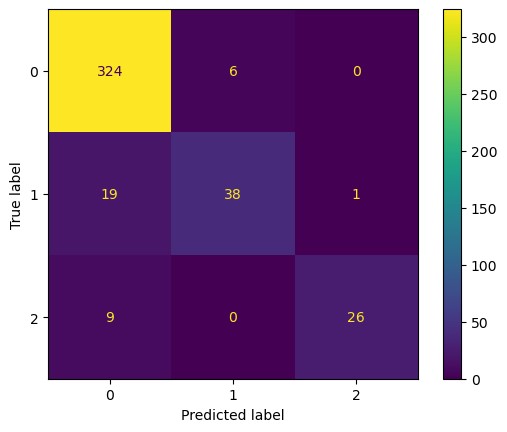
### ADABOOST CLASSIFIER

**from** sklearn.ensemble **import** AdaBoostClassifier model = AdaBoostClassifier(n\_estimators=250, learning\_rate=0.1) model.fit(X\_train, y\_train)

print("For the amounts of training data is: ",size) print("Accuracy of AdaBoost:",model.score(X\_test, y\_test)) y\_pred = model.predict(X\_test) cm = confusion\_matrix(y\_test, y\_pred) cm\_display = ConfusionMatrixDisplay(cm).plot() plt.show()

For the amounts of training data is: 1689

Accuracy of AdaBoost: 0.91725768321513



model = LogisticRegression(max\_iter=1000, C=0.01, penalty="l2", solver="newton-cg") model.fit(X\_train, y\_train) print("For the C is : 0.01 ,Accuracy : ",model.score(X\_test,y\_test))

model = LogisticRegression(max\_iter=1000, C=0.001, penalty="l2", solver="newton-cg") model.fit(X\_train, y\_train) print("For the C is : 0.001 ,Accuracy : ",model.score(X\_test,y\_test))

model = LogisticRegression(max\_iter=1000, C=0.0001, penalty="l2", solver="newton-cg") model.fit(X\_train, y\_train) print("For the C is : 0.0001 ,Accuracy : ",model.score(X\_test,y\_test))

For the C is : 0.01 ,Accuracy : 0.8747044917257684

For the C is : 0.001 ,Accuracy : 0.8723404255319149

For the C is : 0.0001 ,Accuracy : 0.851063829787234

model = KNeighborsClassifier(n\_neighbors=3) model.fit(X\_train, y\_train) print("For the n\_neighbors is 3, Accuracy :",model.score(X\_test, y\_test))

model = KNeighborsClassifier(n\_neighbors=5) model.fit(X\_train, y\_train)

print("For the n\_neighbors is 5, Accuracy :",model.score(X\_test, y\_test))

model = KNeighborsClassifier(n\_neighbors=7) model.fit(X\_train, y\_train) print("For the n\_neighbors is 7, Accuracy :",model.score(X\_test, y\_test))

For the n\_neighbors is 3, Accuracy : 0.8936170212765957

For the n\_neighbors is 5, Accuracy : 0.8865248226950354

For the n\_neighbors is 7, Accuracy : 0.8770685579196218

X\_train1 = X\_train.iloc[:1400,:] y\_train1 = y\_train.iloc[:1400]

size = X\_train1.shape[0]

model = LogisticRegression(max\_iter=1000, C=0.009, penalty="l2", solver="newton-cg") model.fit(X\_train1, y\_train1)

print("For the amounts of training data is: ",size)

print("Accuracy of LogisticRegression: ",model.score(X\_test,y\_test)) print(" ")

model = DecisionTreeClassifier() model.fit(X\_train1, y\_train1)

print("For the amounts of training data is: ",size) print("Accuracy of DecisionTree: ",model.score(X\_test, y\_test)) print(" ")

model = KNeighborsClassifier(n\_neighbors=5) model.fit(X\_train1, y\_train1)

print("For the amounts of training data is: ",size) print("Accuracy of K-NN:",model.score(X\_test, y\_test)) print(" ")

model = AdaBoostClassifier(n\_estimators=250, learning\_rate=0.1) model.fit(X\_train1, y\_train1)

print("For the amounts of training data is: ",size) print("Accuracy of AdaBoost:",model.score(X\_test, y\_test)) print(" ")

rfc = RandomForestClassifier(n\_estimators=100, random\_state=42) rfc = rfc.fit(X\_train, y\_train)

print("For the amounts of training data is: ",size)

print("Accuracy of RandomForestClassifier :",model.score(X\_test, y\_test)) print(" ")

For the amounts of training data is: 1400

Accuracy of LogisticRegression: 0.7807570977917981

For the amounts of training data is: 1400

Accuracy of DecisionTree: 0.9211356466876972

For the amounts of training data is: 1400

Accuracy of K-NN: 0.9132492113564669

For the amounts of training data is: 1400

Accuracy of AdaBoost: 0.9132492113564669

For the amounts of training data is: 1400

Accuracy of RandomForestClassifier : 0.9132492113564669

(2112, 20)

# CHAPTER 6

## CONCLUSION

Fetal health is a critical aspect of prenatal care, as it directly impacts the health outcomes of both the mother and the developing fetus. Monitoring fetal health through regular medical check-ups and tests is essential for identifying potential risks and managing them appropriately. Early detection of fetal distress or other complications can improve the chances of a healthy pregnancy and delivery, while promoting overall fetal development and health can help ensure the longterm health of the newborn.

Various medical procedures and tests are available to monitor fetal health, including ultrasound examinations, fetal heart rate monitoring, and amniotic fluid volume assessments. In addition, machine learning algorithms and models can be used to predict fetal health and identify potential risks. These tools can assist healthcare professionals in providing personalized care to expectant mothers and can help improve the accuracy of fetal health assessments.

Overall, prioritizing fetal health is an essential component of prenatal care. With proper monitoring and management, potential risks can be identified and addressed, leading to a higher likelihood of a successful pregnancy and delivery. By promoting fetal development and health, healthcare professionals can help ensure that newborns have the best possible chance for a healthy start to life.