IDENTIFYING PERINATAL HEALTH RISK USING MACHINE LEARNING TECHNIQUES

In partial fulfillment award of the degree

of

BACHELOR OF ENGINEERING

IN

BIO MEDICAL ENGINEERING

PROGRESS THROUGH KNOWLEDGE

TRICHY - 621112

ANNA UNIVERSITY: CHENNAI 600025

NOVEMBER 2022

ANNA UNIVERSITY: CHENNAI 600 025

A PROJECT REPORT
Submitted by

S Barath kumar	Team Leader
P.Dhivahar	Team Member
S Shyam ganesh	Team Member 2
M jayaprakash	Team Member 3

DHANALAKSHMI SRINIVSAN INSTITUTE OF THE TECHNOLOGY
TIRUCHIRAPPALLI - 621 105

1. INTRODUCTION :

1.1 Project overview :

Identifying and predicting perinatal health risks is of utmost importance in ensuring the well-being of both the mother and the newborn. Perinatal health refers to the period surrounding childbirth, encompassing prenatal, intrapartum, and postnatal stages. Various factors, such as maternal health history, prenatal care, genetic factors, and complications during pregnancy and delivery, can significantly impact the health outcomes for both the mother and the baby.

Machine learning techniques offer a promising approach to analyzing and leveraging vast amounts of perinatal health data to identify potential risks and make accurate predictions. By employing advanced algorithms and statistical models, machine learning can uncover hidden patterns and relationships within the data, providing valuable insights into perinatal health outcomes.

The use of machine learning techniques in perinatal health risk identification enables healthcare professionals to make informed decisions and interventions, leading to improved maternal and neonatal care. By leveraging large-scale datasets, including electronic health records, medical imaging, and wearable devices, machine learning models can provide timely and accurate predictions, aiding in early detection and prevention of perinatal complications.

The application of machine learning in this domain involves several key steps. First, relevant perinatal health data is collected from various sources and preprocessed to ensure its quality and compatibility with machine learning algorithms. Data cleaning, transformation, and feature engineering techniques are employed to prepare the data for analysis.

Next, machine learning models are developed and trained using labeled data, where the labels indicate the presence or absence of perinatal health risks. These models can encompass a range of algorithms, from traditional ones such

as logistic regression, decision trees, and support vector machines, to more advanced deep learning models like convolutional neural networks or recurrent neural networks.

1.2 PURPOSE :

. Introduction:

Perinatal health is a critical aspect of maternal and child healthcare, and early identification of potential risks is vital for effective intervention and improved outcomes. This project aims to leverage machine learning techniques to develop a robust system for identifying perinatal health risks. By analyzing comprehensive datasets and applying advanced algorithms, the system will provide accurate predictions and assist healthcare professionals in making informed decisions during pregnancy, delivery, and postnatal care.

2. Objectives:

- Develop a system that can effectively collect and preprocess perinatal health data from various sources, ensuring data quality and compatibility.
- Explore and implement feature selection/extraction techniques to identify the most relevant factors influencing perinatal health risks.
- Design and train machine learning models to predict perinatal health risks based on the selected features.
- Evaluate the performance of the models using appropriate metrics to assess accuracy and reliability.
- Deploy the trained models in a user-friendly interface, allowing healthcare professionals to input patient data and receive risk predictions in real-time.
- Ensure the system's scalability, security, and privacy, adhering to data protection regulations and best practices.

3. Methodology:

- Gather a diverse and comprehensive dataset of perinatal health records, including maternal health history, prenatal care information, genetic factors, and complications.
- Preprocess the data by handling missing values, cleaning outliers, and transforming features as necessary to prepare it for machine learning analysis.
- Implement feature selection/extraction techniques to identify the most informative features, reducing dimensionality and enhancing model performance.

- Train and validate machine learning models using appropriate algorithms such as logistic regression, decision trees, random forests, or deep learning models.
- Evaluate model performance using metrics such as accuracy, precision, recall, and F1-score through cross-validation and comparison with clinical guidelines.
- Develop an intuitive user interface that allows healthcare professionals to input patient data and obtain risk predictions seamlessly.
- Ensure the system's scalability to handle increasing amounts of data and provide real-time predictions.
- Incorporate robust security measures to protect sensitive patient information and ensure compliance with data protection regulations.

4. Deliverables:
- A fully functional system capable of collecting, preprocessing, and analyzing perinatal health data.
- Trained machine learning models that accurately predict perinatal health risks.
- Evaluation metrics and results demonstrating the performance of the models.
- A user-friendly interface for healthcare professionals to interact with the system and receive risk predictions.
- Documentation detailing the system's architecture, data preprocessing techniques, model development, and deployment instructions.
- Recommendations for further enhancements and future research in the field of perinatal health risk identification using machine learning.

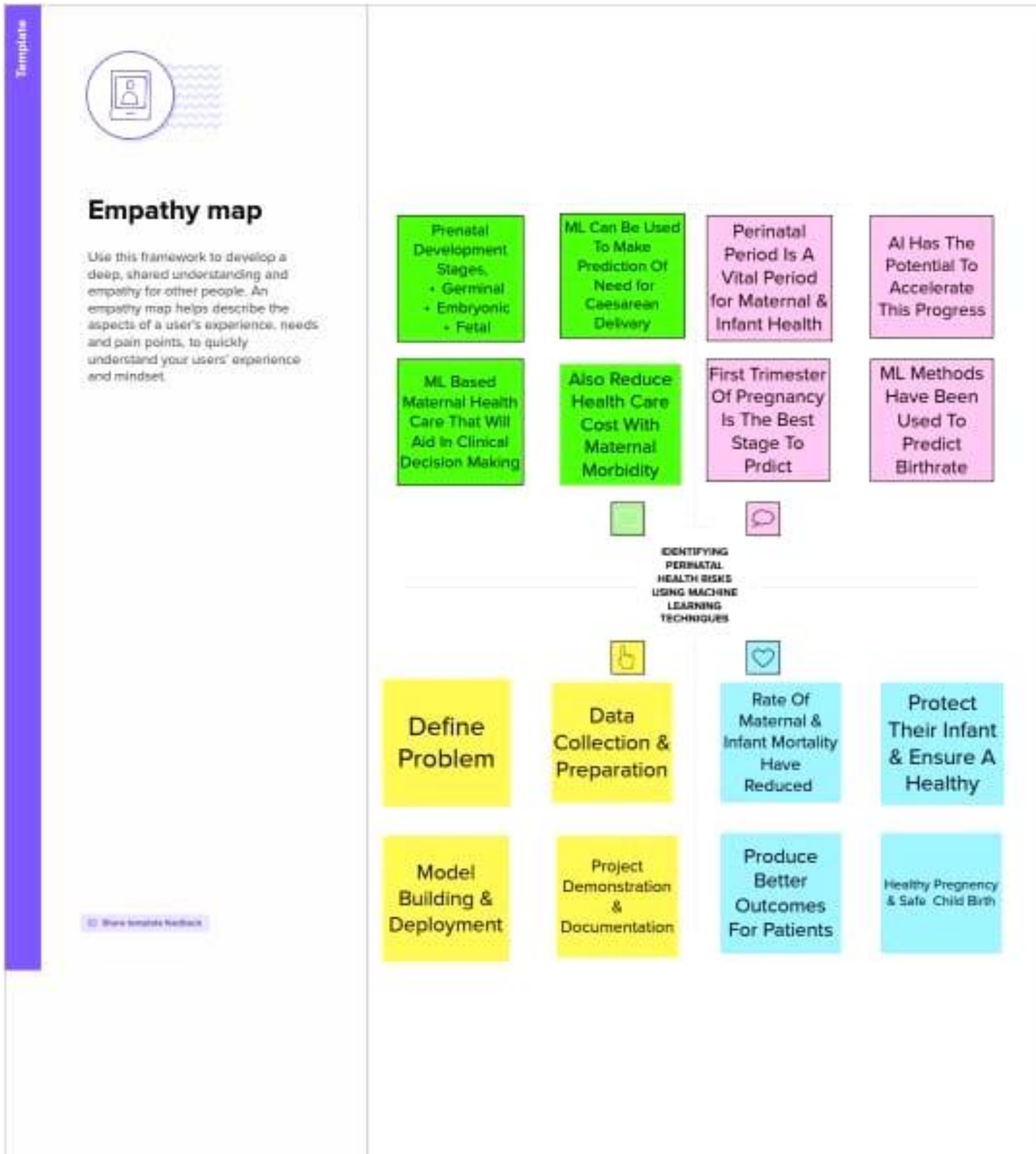
5. Timeline:
- Data collection and preprocessing: 2 months
- Feature selection/extraction and model development: 2 months
- Model training, evaluation, and optimization: 2 months
- User interface development and integration: 1 month
- Documentation and finalization: 1 month

2 . IDEATION &PROPOSED SOLUTION

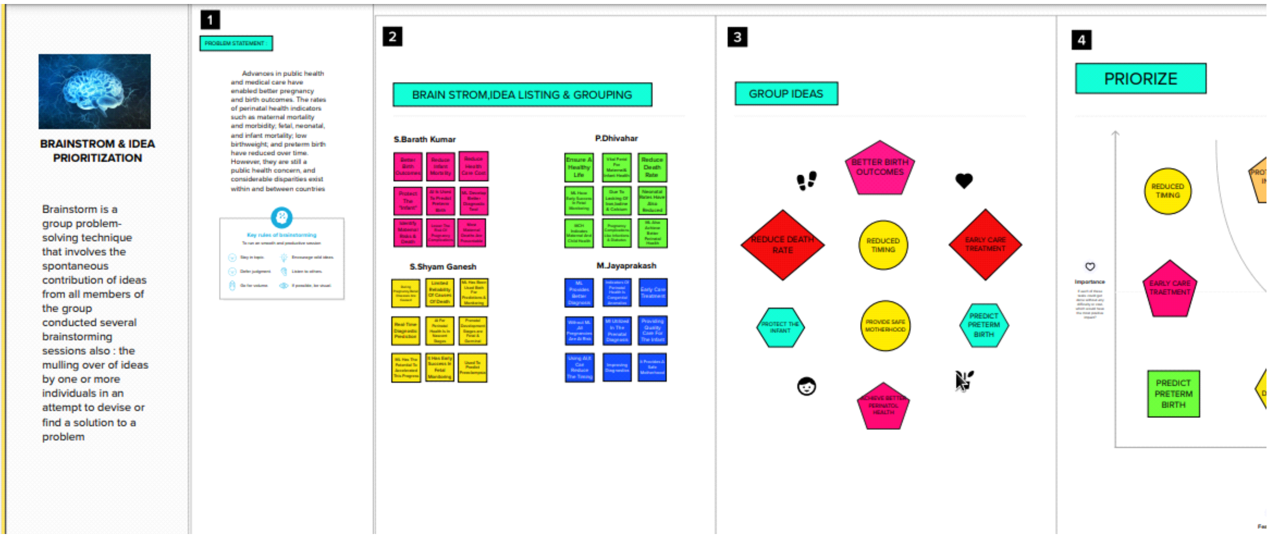
2.1 PROBLEM STATMENT DEFINITION

PROBLEM STATEMENT	I AM (CUSTOMER)	I AM TRYING TO	BUT	BECAUSE	WHICH MAKES ME FEEL
PS-1	Patient	To Overcome The Complications Of Pregnancy	It Is Little Bit Difficult	Becaue Of Lack Of Money & Specialized Care	Frustrated & Depression
PS-2	Doctor	To Summarize The ML Techniques To Predict Perinatal Complications	It Is Little Bit Expensive	It's Difficult To Give Specialized Care To The Care To The Patient	Frustrated & Depression

2.2 EMPATHY MAP CANVAS



2.3 BRAINSTORMING &IDEA PRIORITIZATION



2.4 PROPOSED SOLUTION

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	Identifying Perinatal Health Risks using Machine Learning Techniques
2.	Idea / Solution description	There are no definite ways to prevent many of the leading causes of Perinatal risks. However, there are ways to reduce the risk's. During Pregnancy, a mother should receive early and regular prenatal care. This type of care helps promote the best outcomes for mother and baby
3.	Novelty / Uniqueness	Unique Risk factors in the perinatal period include pregnancy-related complications, prematurity and low birth weight, and infection exposure during pregnancy or at time of birth.
4.	Social Impact / Customer Satisfaction	The loss of an expected child can be devastating and traumatizing for parents, placing them at risk for postloss mental health complications, such as complicated or traumatic grief
5.	Business Model (Revenue Model)	A marketing/business model using non-traditional Quality of Life measures was developed to assess perinatal health status on a micro-geographic level. The Perinatal Region is developing strategies to implement the media usage and consumer behavior marketing information to focus their prevention efforts to the high risk areas in the region based on the Quality of Life Measurements.
6.	Scalability of the Solution	Pregnancy is the most important phase in women's life. There is lot of concern to reduce maternal mortality and infant mortality. Not all birth defects can be prevented. But you can increase your chances of having a healthy baby by managing health conditions and by adopting

3. REQUIREMENT ANALYSIS

3.1 Functional requirements & NON Functional requirementS

Function Requirement:

1. Data Collection: The system should be able to collect relevant perinatal health data, including maternal health history, prenatal care information, genetic factors, and any complications during pregnancy and delivery. This data can be obtained from electronic health records, medical imaging, wearable devices, and other relevant sources.

2. Data Preprocessing: The system should preprocess the collected data to ensure its quality and compatibility with machine learning algorithms. This may involve cleaning the data, handling missing values, normalizing or standardizing features, and removing outliers.

3. Feature Selection/Extraction: The system should identify the most relevant features from the preprocessed data that are likely to have an impact on perinatal health risks. This may involve techniques such as correlation analysis, principal component analysis (PCA), or other feature selection algorithms.

4. Machine Learning Model Development: The system should develop machine learning models that can accurately predict perinatal health risks based on the selected features. Various algorithms can be explored, such as logistic regression, decision trees, random forests, support vector machines (SVM), or deep learning models like convolutional neural networks (CNN) or recurrent neural networks (RNN).

5. Model Training and Evaluation: The system should train the machine learning models using labeled data, where the labels indicate the presence or absence of perinatal health risks. The trained models should be evaluated

using appropriate evaluation metrics, such as accuracy, precision, recall, and F1-score, to assess their performance.

6. Risk Prediction: Once the models are trained and validated, the system should be able to use them to predict perinatal health risks for new, unseen cases. Given the relevant input data, the system should output the predicted risk probability or a binary classification indicating the presence or absence of perinatal health risks.

Non-Functional Requirements:

1. Performance: The system should be able to process and analyze large volumes of perinatal health data efficiently, providing timely predictions and minimizing response times.

2. Accuracy: The machine learning models should strive for high accuracy in predicting perinatal health risks to ensure reliable results.

3. Scalability: The system should be designed to handle an increasing amount of data as more healthcare providers and individuals contribute to the dataset.

4. Security and Privacy: The system should incorporate robust security measures to protect sensitive perinatal health data from unauthorized access or breaches. It should also comply with relevant data protection regulations, ensuring confidentiality and privacy of patient information.

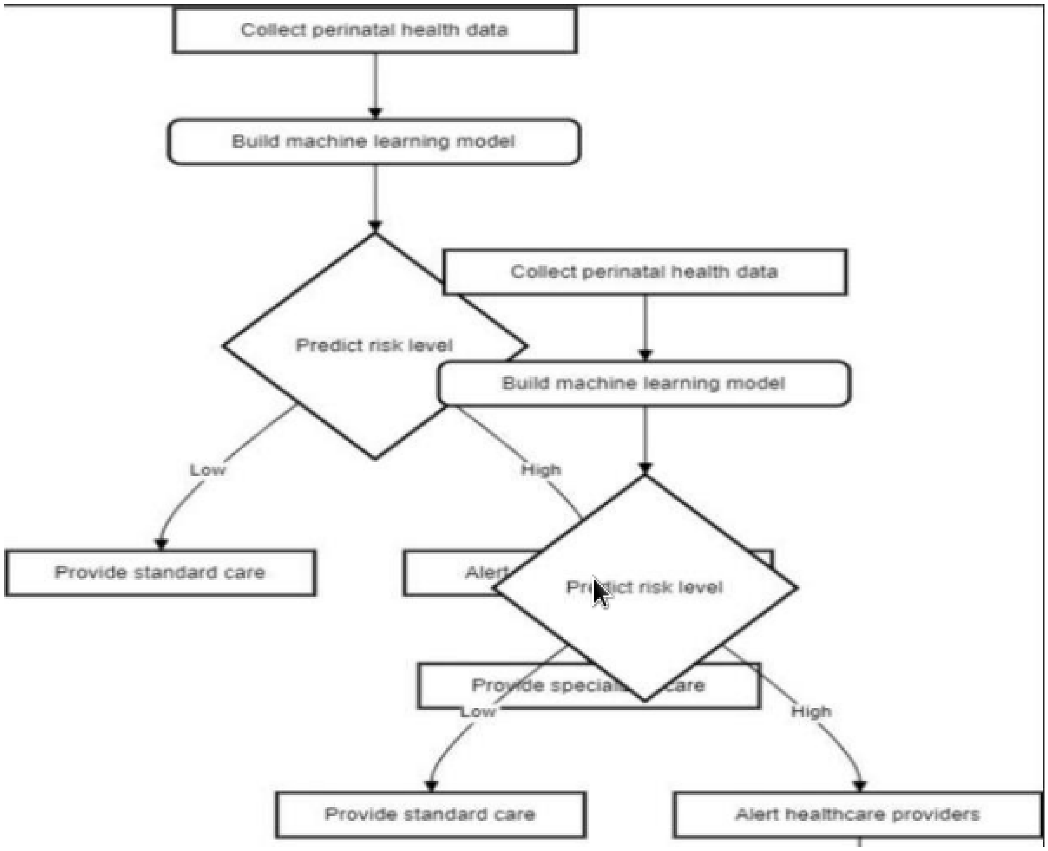
5. Interpretability: The system should provide explanations or interpretability of the predictions made by the machine learning models. This would help healthcare professionals understand the factors contributing to the risk assessment and make informed decisions.

6. Usability: The system should have a user-friendly interface that allows healthcare professionals to input relevant data, interpret results, and interact with the system easily. It should also be accessible to users with varying levels of technical expertise.

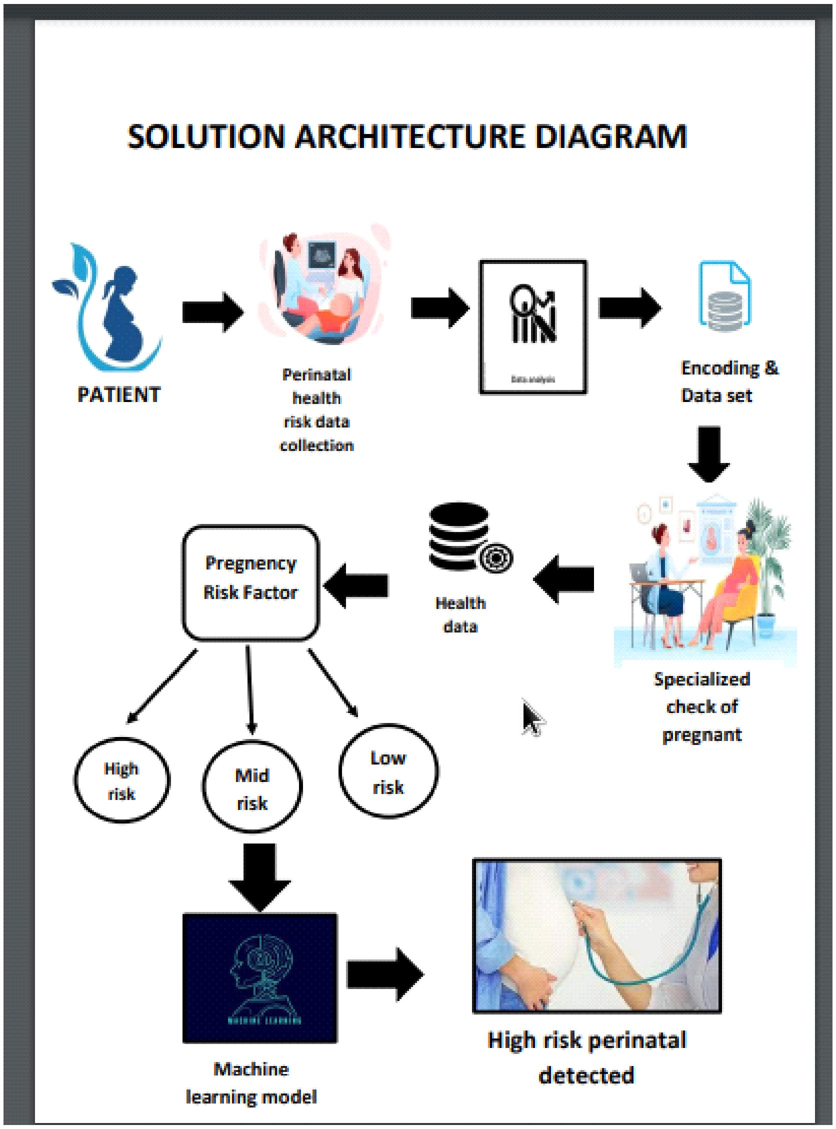
7. Reliability and Availability: The system should be reliable and available for use, minimizing downtime and ensuring continuous access to its functionalities. This may involve redundancy, fault tolerance, and backup mechanisms to handle unexpected failures or disruptions.

4. PROJECT DESIGN

4.1 Data Flow Diagram



4.2 SOLUTION & TECHNICAL ARCHITECTURE :


SOLUTION ARCHITECTURE DIAGRAM
PATIENT
Perinatal health risk data collection
Encoding & Data set
Specialized check of pregnant
Health data
Pregnency Risk Factor
High risk
Mid risk
Low risk
Machine learning model
High risk perinatal detected

5.3 DATABASE SCHEMA

DOWNLOAD THE DATA SET

+ Code + Text

Download the Dataset

```python
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings("ignore")
from sklearn.model_selection import train_test_split as split
from sklearn.model_selection import GridSearchCV
from sklearn.tree import DecisionTreeClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.svm import SVC
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier
from sklearn.ensemble import BaggingClassifier
from sklearn.ensemble import AdaBoostClassifier
from sklearn.naive_bayes import GaussianNB
from sklearn import metrics
from sklearn.metrics import accuracy_score, confusion_matrix
from sklearn.metrics import classification_report
```

Read The Database

```python
# Setting a random seed in order to keep the same random results each time the notebook is run
np.random.seed(seed=11)
```

LOAD THE DATASET

Load the Dataset

```
[140] from google.colab import drive
```

```
[141] drive.mount('/content/drive')
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force_remount=True).
```

```
[142] data =pd.read_csv('/content/drive/MyDrive/Maternal Health Risk Data Set.csv')
```

Handling missing values

```
[148] data.describe()
```

	Age	SystolicBP	DiastolicBP	BS	BodyTemp	HeartRate
count	1014.000000	1014.000000	1014.000000	1014.000000	1014.000000	1014.000000
mean	29.871795	113.198225	76.460552	8.725986	98.665089	74.301775
std	13.474386	18.403913	13.885796	3.293532	1.371384	8.088702
min	10.000000	70.000000	49.000000	6.000000	98.000000	7.000000
25%	19.000000	100.000000	65.000000	6.900000	98.000000	70.000000
50%	26.000000	120.000000	80.000000	7.500000	98.000000	76.000000
75%	39.000000	120.000000	90.000000	8.000000	98.000000	80.000000
max	70.000000	160.000000	100.000000	19.000000	103.000000	90.000000

HANDLING MISSING VALUES

```
[150] print(" number missing the value each column:")
      print(data.isnull().sum())

      data = data.dropna ()

      data= data.fillna(0)

      data = data.fillna (data.mean())

      data = data.fillna(data.mode().iloc[0])

      data= data.fillna (data.median())

      data.to_csv('cleaned_data.csv', index=False)
```

```
 number missing the value each column:
Age            0
SystolicBP     0
DiastolicBP    0
BS             0
BodyTemp       0
HeartRate      0
RiskLevel      0
dtype: int64
```

HANDLING CATEGORICAL DATA

+ Code + Text

Handling categorical data

```
data.info
```

```
<bound method DataFrame.info of       Age  SystolicBP  DiastolicBP    BS  BodyTemp  HeartRate  RiskLevel
0      25         130           80  15.0      98.0         86  high risk
1      35         140           90  13.0      98.0         70  high risk
2      29          90           70   8.0     100.0         80  high risk
3      30         140           85   7.0      98.0         70  high risk
4      35         120           60   6.1      98.0         76   low risk
...   ...         ...          ...   ...       ...        ...        ...
1009   22         120           60  15.0      98.0         80  high risk
1010   55         120           90  18.0      98.0         60  high risk
1011   35          85           60  19.0      98.0         86  high risk
1012   43         120           90  18.0      98.0         70  high risk
1013   32         120           65   6.0     101.0         76   mid risk

[1014 rows x 7 columns]>
```

HANDLING OUTLIER

Handling outliers

```
[156] import numpy as np
      from scipy import stats

      def find_outliers(data):

          z_scores = np.abs(stats.zscore(data))


          threshold = 3


          outlier_indices = np.where(z_scores > threshold)[0]

          return outlier_indices


      data = [2, 4, 5, 7, 8, 10, 12, 14, 15, 100]
      outlier_indices = find_outliers(data)

      print("Outlier indices:", outlier_indices)
```

```
Outlier indices: []
```

Splitting dataset into training and test set

```
[179] import pandas as pd

      # Load the dataset
      data = pd.read_csv('/content/drive/MyDrive/Maternal Health Risk Data Set.csv')

      # Display information about the dataset
      print(data.info())

      # Split the dataset into a training set
      train_set = data.sample(frac=0.8, random_state=42)  # 80% of the data for training

      # Display the information about the training set
      print(train_set.info())
```

SPLITTING DATASET INTO TRAINING AND TEST SET

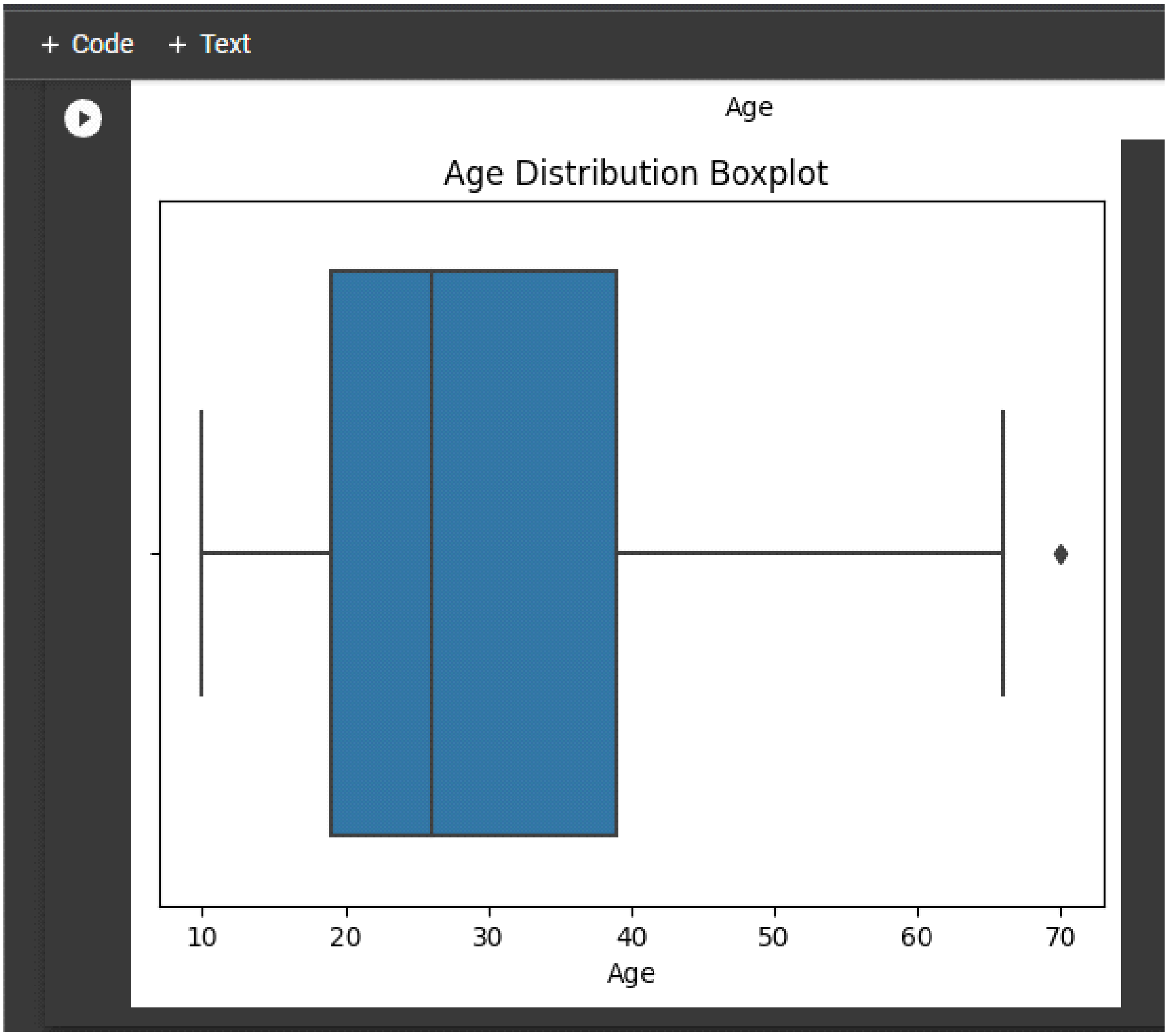
+ Code + Text

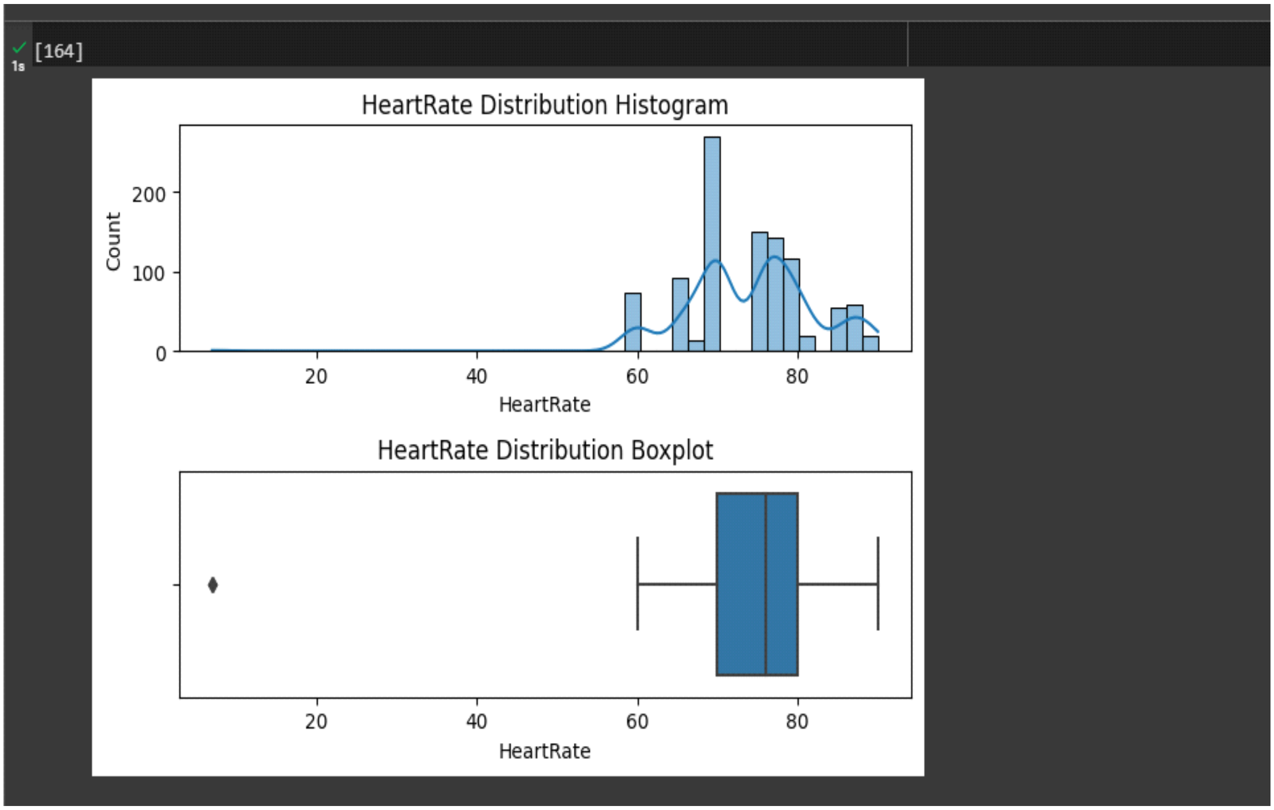
[179]

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1014 entries, 0 to 1013
Data columns (total 7 columns):
 #   Column       Non-Null Count  Dtype  
---  ------       --------------  -----  
 0   Age          1014 non-null   int64  
 1   SystolicBP   1014 non-null   int64  
 2   DiastolicBP  1014 non-null   int64  
 3   BS           1014 non-null   float64
 4   BodyTemp     1014 non-null   float64
 5   HeartRate    1014 non-null   int64  
 6   RiskLevel    1014 non-null   object 
dtypes: float64(2), int64(4), object(1)
memory usage: 55.6+ KB
None
<class 'pandas.core.frame.DataFrame'>
Int64Index: 811 entries, 752 to 645
Data columns (total 7 columns):
 #   Column       Non-Null Count  Dtype  
---  ------       --------------  -----  
 0   Age          811 non-null    int64  
 1   SystolicBP   811 non-null    int64  
 2   DiastolicBP  811 non-null    int64  
 3   BS           811 non-null    float64
 4   BodyTemp     811 non-null    float64
 5   HeartRate    811 non-null    int64  
 6   RiskLevel    811 non-null    object 
dtypes: float64(2), int64(4), object(1)
memory usage: 50.7+ KB
None
```

+ Code + Text

```python
import seaborn as sns
import matplotlib.pyplot as plt

col = 'Age'
Q1 = data[col].quantile(0.25)
Q3 = data[col].quantile(0.75)
IQR = Q3 - Q1

lower_bound = Q1 - 1.5 * IQR
upper_bound = Q3 + 1.5 * IQR

plt.figure(figsize=(8, 6))
sns.boxplot(data=data, x=col)
plt.title(f"{col} Distribution Boxplot")

outliers = data[(data[col] < lower_bound) | (data[col] > upper_bound)]
plt.scatter(x=outliers.index, y=outliers[col], color='red', label='Outliers')
plt.legend()
plt.show()

import seaborn as sns
import matplotlib.pyplot as plt

sns.boxplot(x=df['Age'])
plt.xlabel('Age')
plt.title(' Age Distribution Boxplot')
plt.show()
```

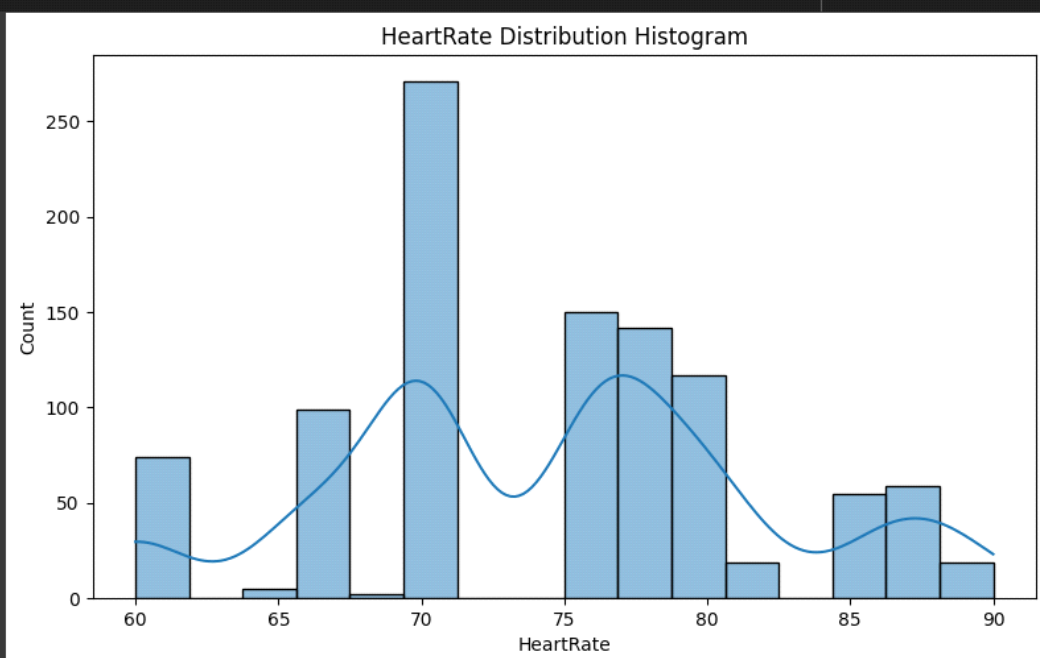

+ Code
+ Text
Age
Age Distribution Boxplot
10
20
30
40
50
60
70
Age


[164]
HeartRate Distribution Histogram
Count
200
100
0
20
40
60
80
HeartRate
HeartRate Distribution Boxplot
20
40
60
80
HeartRate

```
[165] data=data.drop(data.index[data.HeartRate==7])
```

```
[166] import seaborn as sns
      import matplotlib.pyplot as plt

      # Assuming you have a DataFrame called "data" and "col" is the column name you want to plot
      col = 'HeartRate'
      # Create the subplots
      fig, ax = plt.subplots(2, 1, figsize=(8, 10))

      # Create a histogram with kernel density estimation
      sns.histplot(data=data, x=col, kde=True, ax=ax[0])
      ax[0].set_title(f"{col} Distribution Histogram")

      # Create a boxplot
      sns.boxplot(data=data, x=col, ax=ax[1])
      ax[1].set_title(f"{col} Distribution Boxplot")

      # Adjust the spacing between subplots
      plt.tight_layout()

      # Show the plots
      plt.show()
```

0s completed at 9:51 PM



```
[ ]  x=data.drop("Risklevel",axis=1)
     y=data.Risklevel
     x_train,x_test ,y_train, y_test = split(x, y, test_size=0.2, random_state=1)
```

7. ADVANTAGES & DISADVANTAGES :

1. Early Risk Detection: Machine learning models can analyze large amounts of data and identify patterns that may indicate potential perinatal health risks. This enables early detection of risks, allowing healthcare providers to intervene and provide appropriate care in a timely manner, potentially improving outcomes for both the mother and the baby.

2. Personalized Care: Machine learning models can take into account various factors such as maternal medical history, genetic information, and lifestyle factors to provide personalized risk assessments. This can help healthcare providers tailor their care plans and interventions based on the specific needs and risks of individual patients, leading to more targeted and effective treatments.

3. Improved Decision-Making: Machine learning models can assist healthcare providers in making informed decisions by providing evidence-based predictions and recommendations. This can help clinicians prioritize resources and

interventions, optimize care delivery, and potentially reduce the likelihood of adverse events.

4. Scalability and Efficiency: Once developed and trained, machine learning models can process large amounts of data quickly and efficiently, enabling the identification of health risks at scale. This scalability can be particularly beneficial in healthcare systems with high patient volumes, where manual risk assessment may be time-consuming and resource-intensive.

Disadvantages of Identifying Perinatal Health Risks using Machine Learning Techniques:

Data Limitations: Machine learning models rely on high-quality, diverse, and comprehensive datasets to make accurate predictions. However, the availability and quality of perinatal health data may vary, leading to potential biases and limitations in the model's performance. Insufficient or incomplete data can affect the model's ability to identify all relevant risk factors accurately.

CONCLUSION :

In conclusion, identifying perinatal health risks using machine learning techniques holds significant promise in improving perinatal care. Machine learning models have the potential to analyze large amounts of data, detect patterns, and provide personalized risk assessments. This can enable early risk detection, personalized care planning, and evidence-based decision-making, ultimately leading to improved outcomes for pregnant women and their babies.

However, there are several considerations and challenges that need to be addressed. Data limitations, interpretability challenges, ethical concerns, and the need for human-machine collaboration are important factors to consider when implementing machine learning models in perinatal care. It is crucial to ensure the availability of high-quality data, mitigate biases, and establish mechanisms for interpretability and explanation. Additionally, healthcare professionals should view these models as decision support tools, supplementing their expertise rather than replacing it.

By leveraging the strengths of machine learning while addressing the associated limitations, we can harness the potential of these techniques to enhance perinatal care. Continued research, collaboration between healthcare professionals and data scientists, and ongoing evaluation and improvement of the models are essential for successfully integrating machine learning into perinatal care and maximizing its benefits for patients and healthcare providers alike.

FUTURE SCOPE :

The future scope for identifying perinatal health risks using machine learning techniques is promising, with several potential areas of development and advancement:

1. Improved Data Accessibility: As electronic health records become more prevalent and interoperable, there will be increased access to comprehensive perinatal health data. This will enable the development of more robust machine learning models by incorporating a wider range of variables and improving the accuracy of risk predictions.

2. Integration of Wearable Devices and Remote Monitoring: With advancements in wearable devices and remote monitoring technologies, there is an opportunity to collect real-time data on maternal health indicators, fetal well-being, and other relevant factors. Machine learning models can be trained to analyze this continuous streaming data and provide timely risk assessments, enabling proactive interventions and remote monitoring of high-risk pregnancies.

3. Integration of Genomic and Molecular Data: Incorporating genomic and molecular data into machine learning models can provide insights into genetic predispositions and biomarkers associated with perinatal health risks. By integrating this information with clinical data, machine learning models can improve risk stratification and personalized care planning.

4. Predictive Analytics for Complications: Machine learning models can be further developed to predict specific complications such as gestational diabetes, preeclampsia, preterm birth, or fetal growth restriction. By identifying high-risk pregnancies earlier, healthcare providers can intervene promptly and implement preventive strategies, ultimately reducing complications and improving outcomes.

5. Explainable AI and Decision Support Systems: Advancements in explainable artificial intelligence (AI) can enhance the interpretability of machine learning models. This will enable healthcare professionals to understand and trust the

model's predictions, leading to better-informed decision-making and more effective interventions.

6. Integration with Telemedicine and Teleconsultation: Machine learning models can be integrated into telemedicine platforms to provide risk assessments and decision support during remote consultations. This can help extend access to specialized care, especially in underserved areas, and enable remote monitoring and management of high-risk pregnancies.

7. Long-term Health Outcomes Prediction: Machine learning models can be developed to predict long-term health outcomes for both the mother and the baby beyond the perinatal period. By considering a broader range of variables and longitudinal data, these models can provide insights into the long-term implications of perinatal health risks and support proactive interventions and follow-up care.

8. Ethical and Bias Mitigation: Continued efforts are needed to address ethical considerations and mitigate biases in machine learning models. This includes ensuring diverse and representative datasets, considering fairness and equity in risk assessments, and developing transparent and accountable algorithms to minimize the potential for bias and discrimination.

As research and technological advancements progress, machine learning techniques have the potential to revolutionize perinatal care by enabling more precise risk assessments, personalized interventions, and improved outcomes for both mothers and babies. However, it is essential to approach these advancements with caution, considering the ethical, legal, and social implications and ensuring that healthcare professionals remain central to the decision-making process.

APPENDIX :

Appendix: Example Machine Learning Algorithms for Identifying Perinatal Health Risks

Here are a few examples of machine learning algorithms that can be used for identifying perinatal health risks:

1. Logistic Regression: Logistic regression is a commonly used algorithm for binary classification problems, where the goal is to predict the presence or absence of a specific health risk. It can handle both categorical and continuous input variables and provide interpretable results by estimating the probability of a particular outcome.

2. Decision Trees: Decision trees are versatile algorithms that can handle both classification and regression tasks. They create a hierarchical structure of if-else conditions based on input features to make predictions. Decision trees can be particularly useful for identifying complex relationships between multiple risk factors and perinatal outcomes.

3. Random Forests: Random forests are an ensemble learning technique that combines multiple decision trees to make predictions. By aggregating the results of multiple trees, random forests improve accuracy and reduce overfitting. They can handle a large number of input variables and identify important features for risk prediction.

4. Support Vector Machines (SVM): SVM is a powerful algorithm for binary classification tasks. It aims to find an optimal hyperplane that separates different classes with a maximal margin. SVM can handle high-dimensional data and is effective in cases where the data is not linearly separable.

5. Neural Networks: Neural networks, particularly deep learning models, have shown great potential in various healthcare applications. They can capture complex relationships in the data by mimicking the structure and function of the human brain. Neural networks can be used for both classification and regression tasks and have achieved state-of-the-art results in certain perinatal health risk prediction tasks.

source code

base.html

Preview Code Blame 2282 lines (2282 loc) · 736 KB

```
      "    </tr>\n",
      "  </tbody>\n",
      "</table>\n",
      "</div>"
     ],
     "text/plain": [
      "               Age    SystolicBP  DiastolicBP          BS     BodyTemp  \\\n",
      "count  1014.000000  1014.000000  1014.000000  1014.000000  1014.000000   \n",
      "mean     29.871795   113.198225    76.460552     8.725986    98.665089   \n",
      "std      13.474386    18.403913    13.885796     3.293532     1.371384   \n",
      "min      10.000000    70.000000    49.000000     6.000000    98.000000   \n",
      "25%      19.000000   100.000000    65.000000     6.900000    98.000000   \n",
      "50%      26.000000   120.000000    80.000000     7.500000    98.000000   \n",
      "75%      39.000000   120.000000    90.000000     8.000000    98.000000   \n",
      "max      70.000000   160.000000   100.000000    19.000000   103.000000   \n",
      "\n",
      "         HeartRate  \n",
      "count  1014.000000  \n",
      "mean     74.301775  \n",
      "std       8.088702  \n",
      "min       7.000000  \n",
      "25%      70.000000  \n",
```

Preview Code Blame 2282 lines (2282 loc) · 736 KB


```
          "50%      26.000000   120.000000    80.000000     7.500000    98.000000   \n",
          "75%      39.000000   120.000000    90.000000     8.000000    98.000000   \n",
          "max      70.000000   160.000000   100.000000    19.000000   103.000000   \n",
          "\n",
          "         HeartRate  \n",
          "count  1014.000000  \n",
          "mean     74.301775  \n",
          "std       8.088702  \n",
          "min       7.000000  \n",
          "25%      70.000000  \n",
          "50%      76.000000  \n",
          "75%      80.000000  \n",
          "max      90.000000  "
         ]
        },
        "execution_count": 5,
        "metadata": {},
        "output_type": "execute_result"
       }
      ],
      "source": [
       "data.describe()"
```

Preview Code Blame 2282 lines (2282 loc) · 736 KB


```
     },
     "execution_count": 5,
     "metadata": {},
     "output_type": "execute_result"
    }
   ],
   "source": [
    "data.describe()"
   ]
  },
  {
   "cell_type": "markdown",
   "id": "357c9a50",
   "metadata": {},
   "source": [
    "# Checking the dataset distribution"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 6,
   "id": "a55c696d",
```

Preview Code Blame 2282 lines (2282 loc) · 736 KB

```
    "    </tr>\n",
    "    <tr>\n",
    "      <th>25%</th>\n",
    "      <td>19.000000</td>\n",
    "      <td>100.000000</td>\n",
    "      <td>65.000000</td>\n",
    "      <td>6.900000</td>\n",
    "      <td>98.000000</td>\n",
    "      <td>70.000000</td>\n",
    "    </tr>\n",
    "    <tr>\n",
    "      <th>50%</th>\n",
    "      <td>26.000000</td>\n",
    "      <td>120.000000</td>\n",
    "      <td>80.000000</td>\n",
    "      <td>7.500000</td>\n",
    "      <td>98.000000</td>\n",
    "      <td>76.000000</td>\n",
    "    </tr>\n",
    "    <tr>\n",
    "      <th>75%</th>\n",
    "      <td>39.000000</td>\n",
```

Preview | Code | Blame   2282 lines (2282 loc) · 736 KB

```
"      <th></th>\n",
"      <th>Age</th>\n",
"      <th>SystolicBP</th>\n",
"      <th>DiastolicBP</th>\n",
"      <th>BS</th>\n",
"      <th>BodyTemp</th>\n",
"      <th>HeartRate</th>\n",
"    </tr>\n",
"  </thead>\n",
"  <tbody>\n",
"    <tr>\n",
"      <th>count</th>\n",
"      <td>1014.000000</td>\n",
"      <td>1014.000000</td>\n",
"      <td>1014.000000</td>\n",
"      <td>1014.000000</td>\n",
"      <td>1014.000000</td>\n",
"      <td>1014.000000</td>\n",
"    </tr>\n",
"    <tr>\n",
"      <th>mean</th>\n",
"      <td>29.871795</td>\n",
"      <td>113.198225</td>\n",
```

Preview Code Blame 2282 lines (2282 loc) · 736 KB


```
      {
       "data": {
        "text/plain": [
         "array(['high risk', 'low risk', 'mid risk'], dtype=object)"
        ]
       },
       "execution_count": 6,
       "metadata": {},
       "output_type": "execute_result"
      }
     ],
     "source": [
      "data['RiskLevel'].unique()"
     ]
    },
    {
     "cell_type": "code",
     "execution_count": 7,
     "id": "96fed378",
     "metadata": {},
     "outputs": [
      {
```

Preview Code Blame 2282 lines (2282 loc) · 736 KB

```
        "  </tbody>\n",
        "</table>\n",
        "</div>"
       ],
       "text/plain": [
        "   Age  SystolicBP  DiastolicBP     BS  BodyTemp  HeartRate  RiskLevel\n",
        "0   25         130           80  15.00      98.0         86  high risk\n",
        "1   35         140           90  13.00      98.0         70  high risk\n",
        "2   29          90           70   8.00     100.0         80  high risk\n",
        "3   30         140           85   7.00      98.0         70  high risk\n",
        "4   35         120           60   6.10      98.0         76   low risk\n",
        "5   23         140           80   7.01      98.0         70  high risk\n",
        "6   23         130           70   7.01      98.0         78   mid risk\n",
        "7   35          85           60  11.00     102.0         86  high risk\n",
        "8   32         120           90   6.90      98.0         70   mid risk\n",
        "9   42         130           80  18.00      98.0         70  high risk"
       ]
      },
      "execution_count": 3,
      "metadata": {},
      "output_type": "execute_result"
     }
    ],
```

Preview Code Blame 2282 lines (2282 loc) · 736 KB

```
"      <th></th>\n",
"      <th>Age</th>\n",
"      <th>SystolicBP</th>\n",
"      <th>DiastolicBP</th>\n",
"      <th>BS</th>\n",
"      <th>BodyTemp</th>\n",
"      <th>HeartRate</th>\n",
"    </tr>\n",
"  </thead>\n",
"  <tbody>\n",
"    <tr>\n",
"      <th>count</th>\n",
"      <td>1014.000000</td>\n",
"      <td>1014.000000</td>\n",
"      <td>1014.000000</td>\n",
"      <td>1014.000000</td>\n",
"      <td>1014.000000</td>\n",
"      <td>1014.000000</td>\n",
"    </tr>\n",
"    <tr>\n",
"      <th>mean</th>\n",
"      <td>29.871795</td>\n",
"      <td>113.198225</td>\n",
```

Preview Code Blame 2282 lines (2282 loc) · 736 KB


```
  },
  {
   "cell_type": "code",
   "execution_count": 4,
   "id": "ae930de0",
   "metadata": {},
   "outputs": [
    {
     "name": "stdout",
     "output_type": "stream",
     "text": [
      "<class 'pandas.core.frame.DataFrame'>\n",
      "RangeIndex: 1014 entries, 0 to 1013\n",
      "Data columns (total 7 columns):\n",
      " #   Column       Non-Null Count  Dtype  \n",
      "---  ------       --------------  -----  \n",
      " 0   Age          1014 non-null   int64  \n",
      " 1   SystolicBP   1014 non-null   int64  \n",
      " 2   DiastolicBP  1014 non-null   int64  \n",
      " 3   BS           1014 non-null   float64\n",
      " 4   BodyTemp     1014 non-null   float64\n",
      " 5   HeartRate    1014 non-null   int64  \n",
      " 6   RiskLevel    1014 non-null   object \n",
```

Preview Code Blame 2282 lines (2282 loc) · 736 KB


```
       "execution_count": 3,
       "metadata": {},
       "output_type": "execute_result"
      }
     ],
     "source": [
      "data = pd.read_csv('Maternal Health Risk Data Set.csv')\n",
      "data.head(10)"
     ]
    },
    {
     "cell_type": "code",
     "execution_count": 4,
     "id": "ae930de0",
     "metadata": {},
     "outputs": [
      {
       "name": "stdout",
       "output_type": "stream",
       "text": [
        "<class 'pandas.core.frame.DataFrame'>\n",
        "RangeIndex: 1014 entries, 0 to 1013\n",
        "Data columns (total 7 columns):\n",
```

```
    "  </tbody>\n",
    "</table>\n",
    "</div>"
   ],
   "text/plain": [
    "   Age  SystolicBP  DiastolicBP     BS  BodyTemp  HeartRate  RiskLevel\n",
    "0   25         130           80  15.00      98.0         86  high risk\n",
    "1   35         140           90  13.00      98.0         70  high risk\n",
    "2   29          90           70   8.00     100.0         80  high risk\n",
    "3   30         140           85   7.00      98.0         70  high risk\n",
    "4   35         120           60   6.10      98.0         76   low risk\n",
    "5   23         140           80   7.01      98.0         70  high risk\n",
    "6   23         130           70   7.01      98.0         78   mid risk\n",
    "7   35          85           60  11.00     102.0         86  high risk\n",
    "8   32         120           90   6.90      98.0         70   mid risk\n",
    "9   42         130           80  18.00      98.0         70  high risk"
   ]
  },
  "execution_count": 3,
  "metadata": {},
  "output_type": "execute_result"
 }
],
```

```
        "      <td>32</td>\n",
        "      <td>120</td>\n",
        "      <td>90</td>\n",
        "      <td>6.90</td>\n",
        "      <td>98.0</td>\n",
        "      <td>70</td>\n",
        "      <td>mid risk</td>\n",
        "    </tr>\n",
        "    <tr>\n",
        "      <th>9</th>\n",
        "      <td>42</td>\n",
        "      <td>130</td>\n",
        "      <td>80</td>\n",
        "      <td>18.00</td>\n",
        "      <td>98.0</td>\n",
        "      <td>70</td>\n",
        "      <td>high risk</td>\n",
        "    </tr>\n",
        "  </tbody>\n",
        "</table>\n",
        "</div>"
       ],
       "text/plain": [
```

```
"      <td>98.0</td>\n",
"      <td>86</td>\n",
"      <td>high risk</td>\n",
"    </tr>\n",
"    <tr>\n",
"      <th>1</th>\n",
"      <td>35</td>\n",
"      <td>140</td>\n",
"      <td>90</td>\n",
"      <td>13.00</td>\n",
"      <td>98.0</td>\n",
"      <td>70</td>\n",
"      <td>high risk</td>\n",
"    </tr>\n",
"    <tr>\n",
"      <th>2</th>\n",
"      <td>29</td>\n",
"      <td>90</td>\n",
"      <td>70</td>\n",
"      <td>8.00</td>\n",
"      <td>100.0</td>\n",
"      <td>80</td>\n",
"      <td>high risk</td>\n",
```

Preview Code Blame 2282 lines (2282 loc) · 736 KB

```
    "    }\n",
    "</style>\n",
    "<table border=\"1\" class=\"dataframe\">\n",
    "  <thead>\n",
    "    <tr style=\"text-align: right;\">\n",
    "      <th></th>\n",
    "      <th>Age</th>\n",
    "      <th>SystolicBP</th>\n",
    "      <th>DiastolicBP</th>\n",
    "      <th>BS</th>\n",
    "      <th>BodyTemp</th>\n",
    "      <th>HeartRate</th>\n",
    "      <th>RiskLevel</th>\n",
    "    </tr>\n",
    "  </thead>\n",
    "  <tbody>\n",
    "    <tr>\n",
    "      <th>0</th>\n",
    "      <td>25</td>\n",
    "      <td>130</td>\n",
    "      <td>80</td>\n",
    "      <td>15.00</td>\n",
    "      <td>98.0</td>\n",
```

```
"        [1.08566107e-01, 4.51234241e-01, 4.40199652e-01],\n",
"        [1.67432830e-01, 3.57771618e-01, 4.74795553e-01],\n",
"        [8.12664069e-03, 7.21159630e-01, 2.70713729e-01],\n",
"        [2.93594074e-02, 6.59410127e-01, 3.11230465e-01],\n",
"        [1.35068269e-02, 7.99425674e-01, 1.87067499e-01],\n",
"        [3.74504030e-02, 4.08753426e-01, 5.53796171e-01],\n",
"        [4.36180843e-03, 8.44273448e-01, 1.51364743e-01],\n",
"        [2.07997326e-02, 7.46337970e-01, 2.32862297e-01],\n",
"        [1.12023472e-02, 7.95273743e-01, 1.93523910e-01],\n",
"        [3.69538066e-01, 2.25589374e-01, 4.04872561e-01],\n",
"        [1.00271104e-01, 2.82594654e-01, 6.17134241e-01],\n",
"        [2.08841122e-01, 3.27357481e-01, 4.63801397e-01],\n",
"        [1.36083195e-02, 7.76472517e-01, 2.09919163e-01],\n",
"        [9.17903274e-01, 1.89846065e-03, 8.01982650e-02],\n",
"        [1.93339726e-01, 2.02060050e-01, 6.04600224e-01],\n",
"        [7.85751832e-01, 1.78291262e-02, 1.96419041e-01],\n",
"        [9.89569326e-01, 9.78229094e-05, 1.03328514e-02],\n",
"        [2.84787395e-01, 2.22021837e-01, 4.93190768e-01],\n",
"        [4.30707130e-01, 1.32730097e-01, 4.36562774e-01],\n",
"        [1.56857483e-01, 4.72588745e-01, 3.70553772e-01],\n",
"        [5.70537323e-01, 1.16248168e-01, 3.13214509e-01],\n",
"        [4.63427966e-03, 8.33704586e-01, 1.61661134e-01],\n",
"        [1.11313613e-02, 7.99737817e-01, 1.89130822e-01],\n",
```

Preview Code Blame 2282 lines (2282 loc) · 736 KB


```
{
 "cells": [
  {
   "cell_type": "code",
   "execution_count": 2,
   "id": "2b60202d",
   "metadata": {},
   "outputs": [],
   "source": [
    "import pandas as pd\n",
    "import numpy as np\n",
    "import matplotlib.pyplot as plt\n",
    "import seaborn as sns"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 3,
```

Preview Code Blame 2282 lines (2282 loc) · 736 KB

```
            "    </tr>\n",
            "    <tr>\n",
            "      <th>max</th>\n",
            "      <td>70.000000</td>\n",
            "      <td>160.000000</td>\n",
            "      <td>100.000000</td>\n",
            "      <td>19.000000</td>\n",
            "      <td>103.000000</td>\n",
            "      <td>90.000000</td>\n",
            "    </tr>\n",
            "  </tbody>\n",
            "</table>\n",
            "</div>"
           ],
           "text/plain": [
            "               Age   SystolicBP  DiastolicBP           BS     BodyTemp  \\\n",
            "count  1014.000000  1014.000000  1014.000000  1014.000000  1014.000000   \n",
            "mean     29.871795   113.198225    76.460552     8.725986    98.665089   \n",
            "std      13.474386    18.403913    13.885796     3.293532     1.371384   \n",
            "min      10.000000    70.000000    49.000000     6.000000    98.000000   \n",
            "25%      19.000000   100.000000    65.000000     6.900000    98.000000   \n",
            "50%      26.000000   120.000000    80.000000     7.500000    98.000000   \n",
```

```
   "id": "eac2d381",
   "metadata": {},
   "outputs": [],
   "source": [
    "from sklearn.preprocessing import StandardScaler    \n",
    "scale = StandardScaler()  \n",
    "x_train= scale.fit_transform(x_train)    \n",
    "x_test= scale.transform(x_test) "
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 46,
   "id": "62877f54",
   "metadata": {},
   "outputs": [
    {
     "data": {
      "text/plain": [
       "array([[-0.34096191,  0.34432827,  0.9405713 , -0.62741662,  1.73365201,\n",
       "         0.75085067],\n",
       "       [ 0.7672685 ,  2.58609821,  1.67573286,  3.08934025, -0.47759639,\n",
       "         0.38763498],\n",
```

```
      {
       "data": {
        "image/png": "iVBORw0KGgoAAAANSUhEUgAAAVoAAAD4CAYAAACt8i4nAAAAOXRFWHRTb2Z0d2FyZQBNYXRwbG90bGliIHZlcnNpb24zLjMuNCwgaHR0cHM6Ly9tYXRwbG
        "text/plain": [
         "<Figure size 432x288 with 2 Axes>"
        ]
       },
       "metadata": {
        "needs_background": "light"
       },
       "output_type": "display_data"
      }
     ],
     "source": [
      "sns.heatmap(confusion_matrix(y_test,predicted_values),annot=True)"
     ]
    },
    {
     "cell_type": "code",
     "execution_count": 38,
     "id": "e48c3860",
     "metadata": {},
     "outputs": [
```

```
"    <td>35</td>\n",
"    <td>120</td>\n",
"    <td>60</td>\n",
"    <td>6.10</td>\n",
"    <td>98.0</td>\n",
"    <td>76</td>\n",
"    <td>low risk</td>\n",
"   </tr>\n",
"   <tr>\n",
"    <th>5</th>\n",
"    <td>23</td>\n",
"    <td>140</td>\n",
"    <td>80</td>\n",
"    <td>7.01</td>\n",
"    <td>98.0</td>\n",
"    <td>70</td>\n",
"    <td>high risk</td>\n",
"   </tr>\n",
"   <tr>\n",
"    <th>6</th>\n",
"    <td>23</td>\n",
"    <td>130</td>\n",
"    <td>70</td>\n",
```

```
"    }\n",
"</style>\n",
"<table border=\"1\" class=\"dataframe\">\n",
"  <thead>\n",
"    <tr style=\"text-align: right;\">\n",
"      <th></th>\n",
"      <th>Age</th>\n",
"      <th>SystolicBP</th>\n",
"      <th>DiastolicBP</th>\n",
"      <th>BS</th>\n",
"      <th>BodyTemp</th>\n",
"      <th>HeartRate</th>\n",
"      <th>RiskLevel</th>\n",
"    </tr>\n",
"  </thead>\n",
"  <tbody>\n",
"    <tr>\n",
"      <th>0</th>\n",
"      <td>25</td>\n",
"      <td>130</td>\n",
"      <td>80</td>\n",
"      <td>15.00</td>\n",
"      <td>98.0</td>\n",
```

```
     "execution_count": 3,
     "metadata": {},
     "output_type": "execute_result"
    }
   ],
   "source": [
    "data = pd.read_csv('Maternal Health Risk Data Set.csv')\n",
    "data.head(10)"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 4,
   "id": "ae930de0",
   "metadata": {},
   "outputs": [
    {
     "name": "stdout",
     "output_type": "stream",
     "text": [
      "<class 'pandas.core.frame.DataFrame'>\n",
      "RangeIndex: 1014 entries, 0 to 1013\n",
      "Data columns (total 7 columns):\n",
```

```
"   <tr>\n",
"     <th>6</th>\n",
"     <td>23</td>\n",
"     <td>130</td>\n",
"     <td>70</td>\n",
"     <td>7.01</td>\n",
"     <td>98.0</td>\n",
"     <td>78</td>\n",
"     <td>mid risk</td>\n",
"   </tr>\n",
"   <tr>\n",
"     <th>7</th>\n",
"     <td>35</td>\n",
"     <td>85</td>\n",
"     <td>60</td>\n",
"     <td>11.00</td>\n",
"     <td>102.0</td>\n",
"     <td>86</td>\n",
"     <td>high risk</td>\n",
"   </tr>\n",
"   <tr>\n",
"     <th>8</th>\n",
"     <td>32</td>\n"
```

```
    "source": [
     "RF2.score(x_test,y_test)"
    ]
   },
   {
    "cell_type": "code",
    "execution_count": 91,
    "id": "fce87c6f",
    "metadata": {},
    "outputs": [
     {
      "name": "stdout",
      "output_type": "stream",
      "text": [
       "              precision    recall  f1-score   support\n",
       "\n",
       "   high risk       0.87      0.93      0.90        82\n",
       "    low risk       0.89      0.89      0.89       122\n",
       "    mid risk       0.88      0.84      0.86       101\n",
       "\n",
       "    accuracy                           0.88       305\n",
       "   macro avg       0.88      0.88      0.88       305\n",
       "weighted avg       0.88      0.88      0.88       305\n",
```

```
    "      <td>32</td>\n",
    "      <td>120</td>\n",
    "      <td>90</td>\n",
    "      <td>6.90</td>\n",
    "      <td>98.0</td>\n",
    "      <td>70</td>\n",
    "      <td>mid risk</td>\n",
    "    </tr>\n",
    "    <tr>\n",
    "      <th>9</th>\n",
    "      <td>42</td>\n",
    "      <td>130</td>\n",
    "      <td>80</td>\n",
    "      <td>18.00</td>\n",
    "      <td>98.0</td>\n",
    "      <td>70</td>\n",
    "      <td>high risk</td>\n",
    "    </tr>\n",
    "  </tbody>\n",
    "</table>\n",
    "</div>"
   ],
   "text/plain": [
```

```
  },
  "source": [
   "sns.pairplot(data,hue='RiskLevel')"
  ]
 },
 {
  "cell_type": "code",
  "execution_count": 13,
  "id": "22746047",
  "metadata": {},
  "outputs": [
   {
    "name": "stderr",
    "output_type": "stream",
    "text": [
     "/Users/akashjyotiborah/opt/anaconda3/lib/python3.8/site-packages/seaborn/_decorators.py:36: FutureWarning: Pass the following variat
     "  warnings.warn(\n"
    ]
   },
   {
    "data": {
     "text/plain": [
      "<AxesSubplot:xlabel='Age', ylabel='count'>"
```

```
    }},
    "output_type": "display_data"
   }
  ],
  "source": [
   "correlation =data.corr()\n",
   "plt.figure(figsize=(12,8))\n",
   "sns.heatmap(correlation,xticklabels=correlation.columns,yticklabels=correlation.columns,annot =True)"
  ]
 },
 {
  "cell_type": "code",
  "execution_count": 15,
  "id": "45c54a6d",
  "metadata": {},
  "outputs": [
   {
    "name": "stderr",
    "output_type": "stream",
    "text": [
     "/Users/akashjyotiborah/opt/anaconda3/lib/python3.8/site-packages/seaborn/_decorators.py:36: FutureWarning: Pass the following variat
     "  warnings.warn(\n"
    ]
```

```
   {
    "data": {
     "text/plain": [
      "<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x7fb0f0163040>"
     ]
    },
    "execution_count": 56,
    "metadata": {},
    "output_type": "execute_result"
   },
   {
    "data": {
     "image/png": "iVBORw0KGgoAAAANSUhEUgAAAV0AAAEGCAYAAADGwUaDAAAAOXRFWHRTb2Z0d2FyZQBNYXRwbG90bGliIHZlcnNpb24zLjMuNCwgaHR0cHM6Ly9tYXRwbG9
     "text/plain": [
      "<Figure size 432x288 with 2 Axes>"
     ]
    },
    "metadata": {
     "needs_background": "light"
    },
    "output_type": "display_data"
   }
  ],
```

```
 "text/plain": [
  "(array([[1.08519204e-01, 4.38172319e-01, 4.53308477e-01],\n",
  "        [3.67286493e-01, 2.25046426e-01, 4.07667081e-01],\n",
  "        [3.51984465e-02, 6.71670402e-01, 2.93131151e-01],\n",
  "        [1.24790328e-01, 2.69979118e-01, 6.05230554e-01],\n",
  "        [7.50062328e-02, 6.06778974e-01, 3.18214793e-01],\n",
  "        [1.70311995e-01, 1.90044229e-01, 6.39643775e-01],\n",
  "        [7.01688913e-02, 5.63835050e-01, 3.65996059e-01],\n",
  "        [3.54081179e-02, 4.16798732e-01, 5.47793150e-01],\n",
  "        [6.90551802e-01, 4.03908935e-02, 2.69057304e-01],\n",
  "        [5.64562280e-01, 7.21888952e-02, 3.63248825e-01],\n",
  "        [4.38391434e-01, 1.75961733e-01, 3.85646833e-01],\n",
  "        [3.43037566e-01, 2.00922231e-01, 4.56040203e-01],\n",
  "        [2.38763469e-02, 7.09938378e-01, 2.66185275e-01],\n",
  "        [9.65250862e-02, 4.26998473e-01, 4.76476441e-01],\n",
  "        [1.15051179e-01, 4.31354103e-01, 4.53594718e-01],\n",
  "        [1.90148465e-02, 7.66569982e-01, 2.14415171e-01],\n",
  "        [3.74504030e-02, 4.08753426e-01, 5.53796171e-01],\n",
  "        [9.45590561e-03, 8.01446725e-01, 1.89097369e-01],\n",
  "        [3.74504030e-02, 4.08753426e-01, 5.53796171e-01],\n",
  "        [2.97450229e-01, 3.11253517e-01, 3.91296254e-01],\n",
  "        [6.56766625e-01, 2.86051950e-02, 3.14628180e-01],\n",
  "        [9.86290919e-01, 8.43552319e-05, 1.36247256e-02],\n",
```

```
"        [8.81050357e-01, 2.29971805e-03, 1.16649925e-01],\n",
"        [3.87603017e-02, 7.72984243e-01, 1.88255455e-01],\n",
"        [3.30345060e-01, 2.54855755e-01, 4.14799185e-01],\n",
"        [5.64562280e-01, 7.21888952e-02, 3.63248825e-01],\n",
"        [1.09503800e-01, 6.01105588e-01, 2.89390612e-01],\n",
"        [1.76815242e-02, 7.24304609e-01, 2.58013867e-01],\n",
"        [2.17283732e-01, 2.85145026e-01, 4.97571242e-01],\n",
"        [6.81610782e-01, 3.87752319e-02, 2.79613986e-01],\n",
"        [8.31403880e-02, 3.87975336e-01, 5.28884276e-01],\n",
"        [1.39304789e-01, 4.50463890e-01, 4.10231321e-01],\n",
"        [2.15339914e-02, 6.45429918e-01, 3.33036091e-01],\n",
"        [1.36246536e-01, 5.05512767e-01, 3.58240697e-01],\n",
"        [5.05347755e-02, 7.41244042e-01, 2.08221182e-01],\n",
"        [1.13983481e-01, 4.38730601e-01, 4.47285918e-01],\n",
"        [3.54081179e-02, 4.16798732e-01, 5.47793150e-01],\n",
"        [2.34103784e-01, 2.67947145e-01, 4.97949071e-01],\n",
"        [7.17005686e-01, 3.13476771e-02, 2.51646637e-01],\n",
"        [2.10755960e-01, 1.64149074e-01, 6.25094966e-01],\n",
"        [3.78254854e-03, 8.53704092e-01, 1.42513360e-01],\n",
"        [1.35068269e-02, 7.99425674e-01, 1.87067499e-01],\n",
"        [9.68801129e-03, 8.11596747e-01, 1.78715242e-01],\n",
"        [1.25341480e-01, 5.24516632e-01, 3.50141888e-01],\n",
"        [9.00101287e-02, 5.08350212e-01, 4.01639660e-01],\n",
```

```
      ]
     }
    ],
    "source": [
     "print(classification_report(y_test,predict))"
    ]
   },
   {
    "cell_type": "code",
    "execution_count": 87,
    "id": "5ff5d856",
    "metadata": {},
    "outputs": [],
    "source": [
     "RF2= RandomForestClassifier(criterion='entropy',max_depth=30,max_features='auto',n_estimators=500)"
    ]
   },
   {
    "cell_type": "code",
    "execution_count": 88,
    "id": "917b8170",
    "metadata": {},
    "outputs": [
```

```
  },
  "source": [
   "X.shape"
  ]
 },
 {
  "cell_type": "code",
  "execution_count": 11,
  "id": "1db7d872",
  "metadata": {},
  "outputs": [
   {
    "data": {
     "text/plain": [
      "(1014,)"
     ]
    },
    "execution_count": 11,
    "metadata": {},
    "output_type": "execute_result"
   }
  ],
  "source": [
```

```
"        [1.30417069e-02, 7.41424733e-01, 2.45533560e-01],\n",
"        [1.51079559e-01, 4.69419086e-01, 3.79501355e-01],\n",
"        [9.87151825e-01, 1.46750035e-04, 1.27014253e-02],\n",
"        [9.43284300e-01, 1.23688439e-03, 5.54788155e-02],\n",
"        [6.04799405e-01, 6.15430221e-02, 3.33657573e-01],\n",
"        [3.35612905e-01, 2.09318229e-01, 4.55068866e-01],\n",
"        [9.52576205e-01, 6.35260430e-04, 4.67885350e-02],\n",
"        [1.69007785e-03, 9.09452657e-01, 8.88572653e-02],\n",
"        [5.83746038e-01, 6.36970608e-02, 3.52556901e-01],\n",
"        [2.80514121e-01, 2.71847226e-01, 4.47638653e-01],\n",
"        [1.51782056e-01, 5.24056548e-01, 3.24161396e-01],\n",
"        [6.15879037e-01, 5.80376383e-02, 3.26083325e-01],\n",
"        [1.42825753e-01, 3.93447447e-01, 4.63726800e-01],\n",
"        [3.54081179e-02, 4.16798732e-01, 5.47793150e-01],\n",
"        [5.70537323e-01, 1.16248168e-01, 3.13214509e-01],\n",
"        [5.59925142e-02, 5.71830599e-01, 3.72176887e-01],\n",
"        [1.73618720e-01, 4.48609405e-01, 3.77771874e-01],\n",
"        [1.63761674e-02, 7.33420604e-01, 2.50203228e-01],\n",
"        [6.81481161e-01, 3.97506814e-02, 2.78768158e-01],\n",
"        [1.24790328e-01, 2.69979118e-01, 6.05230554e-01],\n",
"        [4.55206232e-01, 2.21517379e-02, 5.22642030e-01],\n",
"        [2.00276414e-02, 7.84583080e-01, 1.95389278e-01],\n",
"        [8.49850960e-01, 8.67209350e-03, 1.41476946e-01],\n",
```

```
"         [1.10242696e-01, 2.65149505e-01, 6.24607800e-01],\n",
"         [5.64562280e-01, 7.21888952e-02, 3.63248825e-01],\n",
"         [5.29128125e-01, 6.96946977e-02, 4.01177178e-01],\n",
"         [1.27441672e-01, 4.09777406e-01, 4.62780922e-01],\n",
"         [2.20946506e-01, 3.22409751e-01, 4.56643743e-01],\n",
"         [5.96585010e-01, 6.43262592e-02, 3.39088731e-01],\n",
"         [2.67552890e-02, 6.74704460e-01, 2.98540251e-01],\n",
"         [2.42203873e-01, 2.47603979e-01, 5.10192148e-01],\n",
"         [4.36868312e-01, 5.16375077e-02, 5.11494180e-01],\n",
"         [5.84295595e-02, 7.56384715e-01, 1.85185725e-01],\n",
"         [1.07436162e-01, 5.54116012e-01, 3.38447826e-01],\n",
"         [1.11848923e-02, 7.47952269e-01, 2.40862838e-01],\n",
"         [6.95611692e-01, 6.01519507e-02, 2.44236357e-01],\n",
"         [2.34103784e-01, 2.67947145e-01, 4.97949071e-01],\n",
"         [3.46891811e-02, 6.67418184e-01, 2.97892634e-01],\n",
"         [6.87919568e-01, 4.10043111e-02, 2.71076121e-01],\n",
"         [9.28490274e-02, 4.72132395e-01, 4.35018578e-01],\n",
"         [7.79654150e-01, 2.34375917e-02, 1.96908259e-01],\n",
"         [1.20747769e-01, 6.44299100e-01, 2.34953131e-01],\n",
"         [1.21302815e-01, 4.46827340e-01, 4.31869846e-01],\n",
"         [3.91494453e-01, 6.56299630e-02, 5.42875583e-01],\n",
"         [6.13831730e-03, 8.12206889e-01, 1.81654793e-01],\n",
"         [8.36800221e-02, 4.99214426e-01, 4.17105552e-01],\n",
```

```
      "<Figure size 432x288 with 1 Axes>"
     ]
    },
    "metadata": {
     "needs_background": "light"
    },
    "output_type": "display_data"
   }
  ],
  "source": [
   "sns.scatterplot('RiskLevel','BS',hue='RiskLevel',data=data)"
  ]
 },
 {
  "cell_type": "code",
  "execution_count": 17,
  "id": "23c1ad69",
  "metadata": {},
  "outputs": [
   {
    "name": "stderr",
    "output_type": "stream",
```

```
    "needs_background": "light"
   },
   "output_type": "display_data"
  },
  {
   "data": {
    "image/png": "iVBORw0KGgoAAAANSUhEUgAAAW4AAAF+CAYAAACidPAUAAAAOXRFWHRTb2Z0d2FyZQBNYXRwbG90bGliIHZlcnNpb24zLjMuNCwgaHR0cHM6Ly9tYXRwbG
    "text/plain": [
     "<Figure size 360x360 with 1 Axes>"
    ]
   },
   "metadata": {
    "needs_background": "light"
   },
   "output_type": "display_data"
  }
 ],
 "source": [
  "sns.catplot(x=\"RiskLevel\", y=\"SystolicBP\", data=data, kind=\"box\").set(title=\"Distribution based on SystolicBP\")\n",
  "sns.catplot(x=\"RiskLevel\", y=\"DiastolicBP\", data=data, kind=\"box\").set(title=\"Distribution based on DiastolicBP\")"
 ]
},
```

```
   },
   "execution_count": 11,
   "metadata": {},
   "output_type": "execute_result"
  }
 ],
 "source": [
  "Y.shape"
 ]
},
{
 "cell_type": "code",
 "execution_count": 12,
 "id": "ca0f2435",
 "metadata": {},
 "outputs": [
  {
   "data": {
    "text/plain": [
     "<seaborn.axisgrid.PairGrid at 0x7fb0e877f9d0>"
    ]
   },
   "execution_count": 12,
```

```
  ],
  "source": [
   "data['RiskLevel'].unique()"
  ]
 },
 {
  "cell_type": "code",
  "execution_count": 7,
  "id": "96fed378",
  "metadata": {},
  "outputs": [
   {
    "data": {
     "text/plain": [
      "low risk     0.400394\n",
      "mid risk     0.331361\n",
      "high risk    0.268245\n",
      "Name: RiskLevel, dtype: float64"
     ]
    },
    "execution_count": 7,
    "metadata": {},
```

```
  "cell_type": "code",
  "execution_count": 9,
  "id": "f53e68fc",
  "metadata": {},
  "outputs": [],
  "source": [
   "X=data.drop(columns=['RiskLevel'])\n",
   "Y=data['RiskLevel']"
  ]
 },
 {
  "cell_type": "code",
  "execution_count": 10,
  "id": "5ebc449f",
  "metadata": {},
  "outputs": [
   {
    "data": {
     "text/plain": [
      "(1014, 6)"
     ]
    },
    "execution_count": 10,
```

```
      "    mid risk       0.79      0.79      0.79       101\n",
      "\n",
      "    accuracy                           0.84       305\n",
      "   macro avg       0.84      0.85      0.84       305\n",
      "weighted avg       0.84      0.84      0.84       305\n",
      "\n"
     ]
    }
   ],
   "source": [
    "print(classification_report(y_test,pred))"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 45,
   "id": "eac2d381",
   "metadata": {},
   "outputs": [],
   "source": [
    "from sklearn.preprocessing import StandardScaler    \n",
    "scale = StandardScaler()  \n",
    "x_train= scale.fit_transform(x_train)    \n",
```

```
"        [1.99310330e-02, 7.09671009e-01, 2.70397958e-01],\n",
"        [9.68801129e-03, 8.11596747e-01, 1.78715242e-01],\n",
"        [9.87211652e-01, 9.90729504e-05, 1.26892752e-02],\n",
"        [1.84630773e-01, 1.76688071e-01, 6.38681156e-01],\n",
"        [2.27441846e-03, 8.83813478e-01, 1.13912104e-01],\n",
"        [6.71992414e-01, 8.58678089e-02, 2.42139777e-01],\n",
"        [3.74504030e-02, 4.08753426e-01, 5.53796171e-01],\n",
"        [1.66174250e-01, 3.60662936e-01, 4.73162814e-01],\n",
"        [5.18253815e-01, 4.42590382e-02, 4.37487147e-01],\n",
"        [9.48182512e-01, 9.87808494e-04, 5.08296798e-02],\n",
"        [9.52238554e-01, 1.70283199e-03, 4.60586139e-02],\n",
"        [9.30458564e-01, 3.23910929e-03, 6.63023271e-02],\n",
"        [8.18266580e-01, 1.68582551e-02, 1.64875165e-01],\n",
"        [1.13983481e-01, 4.38730601e-01, 4.47285918e-01],\n",
"        [1.03603944e-01, 4.52924129e-01, 4.43471926e-01],\n",
"        [6.47610684e-02, 6.38060830e-01, 2.97178102e-01],\n",
"        [8.31403880e-02, 3.87975336e-01, 5.28884276e-01],\n",
"        [1.15423987e-01, 5.42322355e-01, 3.42253658e-01],\n",
"        [7.79654150e-01, 2.34375917e-02, 1.96908259e-01],\n",
"        [3.46482025e-01, 2.62964220e-01, 3.90553755e-01],\n",
"        [1.70878414e-02, 6.28121380e-01, 3.54790779e-01],\n",
"        [1.26988831e-01, 4.19709076e-01, 4.53302093e-01],\n",
"        [1.08566107e-01, 4.51234241e-01, 4.40199652e-01],\n",
```

```
   {
    "data": {
     "image/png": "iVBORw0KGgoAAAANSUhEUgAAAc8AAAHBCAYAAAAPcjBbAAAAOXRFWHRTb2Z0d2FyZQBNYXRwbG90bGliIHZlcnNpb24zLjMuNCwgaHR0cHM6Ly9tYXRwbG9
     "text/plain": [
      "<Figure size 576x576 with 1 Axes>"
     ]
    },
    "metadata": {},
    "output_type": "display_data"
   }
  ],
  "source": [
   "data['RiskLevel'].value_counts().plot.pie(autopct='%.2f',figsize=(8,8))"
  ]
 },
 {
  "cell_type": "markdown",
  "id": "90d70856",
  "metadata": {},
  "source": [
   "# Seperating dependent and independent variable"
  ]
 },
```

```
  "id": "448303d3",
  "metadata": {},
  "outputs": [
   {
    "data": {
     "text/plain": [
      "((709, 6), (305, 6), (709,), (305,))"
     ]
    },
    "execution_count": 23,
    "metadata": {},
    "output_type": "execute_result"
   }
  ],
  "source": [
   "from sklearn.model_selection import train_test_split as tts\n",
   "x_train,x_test,y_train,y_test=tts(scaled_X,Y,train_size=0.70,stratify=Y,random_state=101)\n",
   "x_train.shape,x_test.shape,y_train.shape,y_test.shape"
  ]
 },
 {
  "cell_type": "code",
  "execution_count": 24,
```

```
"        [3.69538066e-01, 2.25589374e-01, 4.04872561e-01],\n",
"        [1.00271104e-01, 2.82594654e-01, 6.17134241e-01],\n",
"        [2.08841122e-01, 3.27357481e-01, 4.63801397e-01],\n",
"        [1.36083195e-02, 7.76472517e-01, 2.09919163e-01],\n",
"        [9.17903274e-01, 1.89846065e-03, 8.01982650e-02],\n",
"        [1.93339726e-01, 2.02060050e-01, 6.04600224e-01],\n",
"        [7.85751832e-01, 1.78291262e-02, 1.96419041e-01],\n",
"        [9.89569326e-01, 9.78229094e-05, 1.03328514e-02],\n",
"        [2.84787395e-01, 2.22021837e-01, 4.93190768e-01],\n",
"        [4.30707130e-01, 1.32730097e-01, 4.36562774e-01],\n",
"        [1.56857483e-01, 4.72588745e-01, 3.70553772e-01],\n",
"        [5.70537323e-01, 1.16248168e-01, 3.13214509e-01],\n",
"        [4.63427966e-03, 8.33704586e-01, 1.61661134e-01],\n",
"        [1.11313613e-02, 7.99737817e-01, 1.89130822e-01],\n",
"        [1.30417069e-02, 7.41424733e-01, 2.45533560e-01],\n",
"        [1.51079559e-01, 4.69419086e-01, 3.79501355e-01],\n",
"        [9.87151825e-01, 1.46750035e-04, 1.27014253e-02],\n",
"        [9.43284300e-01, 1.23688439e-03, 5.54788155e-02],\n",
"        [6.04799405e-01, 6.15430221e-02, 3.33657573e-01],\n",
"        [3.35612905e-01, 2.09318229e-01, 4.55068866e-01],\n",
"        [9.52576205e-01, 6.35260430e-04, 4.67885350e-02],\n",
"        [1.69007785e-03, 9.09452657e-01, 8.88572653e-02],\n",
"        [5.83746038e-01, 6.36970608e-02, 3.52556901e-01],\n",
```

```
      "data": {
       "image/png": "iVBORw0KGgoAAAANSUhEUgAAAV0AAAEGCAYAAADGwUaDAAAAOXRFWHRTb2Z0d2FyZQBNYXRwbG90bGliIHZlcnNpb24zLjMuNCwgaHR0cHM6Ly9tYXRwbG9
       "text/plain": [
        "<Figure size 432x288 with 2 Axes>"
       ]
      },
      "metadata": {
       "needs_background": "light"
      },
      "output_type": "display_data"
     }
    ],
    "source": [
     "plot_confusion_matrix(RF,x_test,y_test)"
    ]
   },
   {
    "cell_type": "code",
    "execution_count": 57,
    "id": "9b5bfe36",
    "metadata": {},
    "outputs": [
     {
```

```
  "metadata": {},
  "outputs": [
   {
    "data": {
     "text/plain": [
      "((709, 6), (305, 6), (709,), (305,))"
     ]
    },
    "execution_count": 23,
    "metadata": {},
    "output_type": "execute_result"
   }
  ],
  "source": [
   "from sklearn.model_selection import train_test_split as tts\n",
   "x_train,x_test,y_train,y_test=tts(scaled_X,Y,train_size=0.70,stratify=Y,random_state=101)\n",
   "x_train.shape,x_test.shape,y_train.shape,y_test.shape"
  ]
 },
 {
  "cell_type": "code",
  "execution_count": 24,
```

```
    "metadata": {},
    "output_type": "execute_result"
   }
  ],
  "source": [
   "from sklearn.linear_model import LogisticRegression as LR\n",
   "classifier=LR(class_weight='balanced')\n",
   "classifier.fit(x_train,y_train)\n",
   "predicted_values=classifier.predict(x_test)\n",
   "predicted_probabilities=classifier.predict_proba(x_test)\n",
   "predicted_values\n",
   "predicted_probabilities,predicted_probabilities.shape"
  ]
 },
 {
  "cell_type": "code",
  "execution_count": 27,
  "id": "6a4e22e9",
  "metadata": {},
  "outputs": [
   {
    "data": {
     "text/plain": [
```

```
     "data": {
      "image/png": "iVBORw0KGgoAAAANSUhEUgAAAV0AAAEGCAYAAADGwUaDAAAAOXRFWHRTb2Z0d2FyZQBNYXRwbG90bGliIHZlcnNpb24zLjMuNCwgaHR0cHM6Ly9tYXRwbG
      "text/plain": [
       "<Figure size 432x288 with 2 Axes>"
      ]
     },
     "metadata": {
      "needs_background": "light"
     },
     "output_type": "display_data"
    }
   ],
   "source": [
    "plot_confusion_matrix(RF,x_test,y_test)"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 57,
   "id": "9b5bfe36",
   "metadata": {},
   "outputs": [
    {
```

```
   },
   "execution_count": 13,
   "metadata": {},
   "output_type": "execute_result"
  },
  {
   "data": {
    "image/png": "iVBORw0KGgoAAAANSUhEUgAAA3UAAAFzCAYAAACZyCAUAAAAOXRFWHRTb2Z0d2FyZQBNYXRwbG90bGliIHZlcnNpb24zLjMuNCwgaHR0cHM6Ly9tYXRwbG9
    "text/plain": [
     "<Figure size 1080x432 with 1 Axes>"
    ]
   },
   "metadata": {
    "needs_background": "light"
   },
   "output_type": "display_data"
  }
 ],
 "source": [
  "plt.figure(figsize=(15,6))\n",
  "sns.countplot('Age',hue='RiskLevel',data=data)"
 ]
},
```

```
      "text/plain": [
       "(305,)"
      ]
     },
     "execution_count": 28,
     "metadata": {},
     "output_type": "execute_result"
    }
   ],
   "source": [
    "predicted_values.shape"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 29,
   "id": "c81a796f",
   "metadata": {},
   "outputs": [
    {
     "data": {
      "text/plain": [
       "(305,)"
```

```
 {
  "cell_type": "code",
  "execution_count": 56,
  "id": "cf72f282",
  "metadata": {},
  "outputs": [
   {
    "name": "stderr",
    "output_type": "stream",
    "text": [
     "/Users/akashjyotiborah/opt/anaconda3/lib/python3.8/site-packages/sklearn/utils/deprecation.py:87: FutureWarning: Function plot_conf
     "  warnings.warn(msg, category=FutureWarning)\n"
    ]
   },
   {
    "data": {
     "text/plain": [
      "<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x7fb0f0163040>"
     ]
    },
    "execution_count": 56,
    "metadata": {},
    "output_type": "execute_result"
```

```
   "source": [
    "RF2.score(x_test,y_test)"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 91,
   "id": "fce87c6f",
   "metadata": {},
   "outputs": [
    {
     "name": "stdout",
     "output_type": "stream",
     "text": [
      "              precision    recall  f1-score   support\n",
      "\n",
      "   high risk       0.87      0.93      0.90        82\n",
      "    low risk       0.89      0.89      0.89       122\n",
      "    mid risk       0.88      0.84      0.86       101\n",
      "\n",
      "    accuracy                           0.88       305\n",
      "   macro avg       0.88      0.88      0.88       305\n",
      "weighted avg       0.88      0.88      0.88       305\n",
```

```
     ]
    }
   ],
   "source": [
    "print(classification_report(y_test,predict))"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 87,
   "id": "5ff5d856",
   "metadata": {},
   "outputs": [],
   "source": [
    "RF2= RandomForestClassifier(criterion='entropy',max_depth=30,max_features='auto',n_estimators=500)"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 88,
   "id": "917b8170",
   "metadata": {},
   "outputs": [
```

Preview Code Blame 2282 lines (2282 loc) · 736 KB

```
          "    </tr>\n",
          "    <tr>\n",
          "      <th>max</th>\n",
          "      <td>70.000000</td>\n",
          "      <td>160.000000</td>\n",
          "      <td>100.000000</td>\n",
          "      <td>19.000000</td>\n",
          "      <td>103.000000</td>\n",
          "      <td>90.000000</td>\n",
          "    </tr>\n",
          "  </tbody>\n",
          "</table>\n",
          "</div>"
         ],
         "text/plain": [
          "               Age   SystolicBP  DiastolicBP           BS     BodyTemp  \\\n",
          "count  1014.000000  1014.000000  1014.000000  1014.000000  1014.000000   \n",
          "mean     29.871795   113.198225    76.460552     8.725986    98.665089   \n",
          "std      13.474386    18.403913    13.885796     3.293532     1.371384   \n",
          "min      10.000000    70.000000    49.000000     6.000000    98.000000   \n",
          "25%      19.000000   100.000000    65.000000     6.900000    98.000000   \n",
          "50%      26.000000   120.000000    80.000000     7.500000    98.000000   \n",
```

```
   "source": [
    "svc_pred=svc.predict(x_test)"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 96,
   "id": "0a5d37ad",
   "metadata": {},
   "outputs": [
    {
     "name": "stdout",
     "output_type": "stream",
     "text": [
      "              precision    recall  f1-score   support\n",
      "\n",
      "   high risk       0.84      0.78      0.81        82\n",
      "    low risk       0.61      0.92      0.73       122\n",
      "    mid risk       0.66      0.29      0.40       101\n",
      "\n",
      "    accuracy                           0.67       305\n",
      "   macro avg       0.70      0.66      0.65       305\n",
      "weighted avg       0.69      0.67      0.64       305\n",
```

```
   "outputs": [],
   "source": [
    "from sklearn.svm import SVC"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 94,
   "id": "20fc6244",
   "metadata": {},
   "outputs": [
    {
     "data": {
      "text/html": [
       "<style>#sk-container-id-11 {color: black;background-color: white;}#sk-container-id-11 pre{padding: 0;}#sk-container-id-11 div.sk-
      ],
      "text/plain": [
       "SVC(kernel='poly')"
      ]
     },
     "execution_count": 94,
     "metadata": {},
     "output_type": "execute_result"
```

Preview Code Blame 2282 lines (2282 loc) · 736 KB

```
            "    </tr>\n",
            "  </tbody>\n",
            "</table>\n",
            "</div>"
           ],
           "text/plain": [
            "             Age   SystolicBP  DiastolicBP           BS     BodyTemp  \\\n",
            "count  1014.000000  1014.000000  1014.000000  1014.000000  1014.000000   \n",
            "mean     29.871795   113.198225    76.460552     8.725986    98.665089   \n",
            "std      13.474386    18.403913    13.885796     3.293532     1.371384   \n",
            "min      10.000000    70.000000    49.000000     6.000000    98.000000   \n",
            "25%      19.000000   100.000000    65.000000     6.900000    98.000000   \n",
            "50%      26.000000   120.000000    80.000000     7.500000    98.000000   \n",
            "75%      39.000000   120.000000    90.000000     8.000000    98.000000   \n",
            "max      70.000000   160.000000   100.000000    19.000000   103.000000   \n",
            "\n",
            "         HeartRate  \n",
            "count  1014.000000  \n",
            "mean     74.301775  \n",
            "std       8.088702  \n",
            "min       7.000000  \n",
            "25%      70.000000  \n",
```

Preview Code Blame 2282 lines (2282 loc) · 736 KB

```
"        [2.34103784e-01, 2.67947145e-01, 4.97949071e-01],\n",
"        [3.46891811e-02, 6.67418184e-01, 2.97892634e-01],\n",
"        [6.87919568e-01, 4.10043111e-02, 2.71076121e-01],\n",
"        [9.28490274e-02, 4.72132395e-01, 4.35018578e-01],\n",
"        [7.79654150e-01, 2.34375917e-02, 1.96908259e-01],\n",
"        [1.20747769e-01, 6.44299100e-01, 2.34953131e-01],\n",
"        [1.21302815e-01, 4.46827340e-01, 4.31869846e-01],\n",
"        [3.91494453e-01, 6.56299630e-02, 5.42875583e-01],\n",
"        [6.13831730e-03, 8.12206889e-01, 1.81654793e-01],\n",
"        [8.36800221e-02, 4.99214426e-01, 4.17105552e-01],\n",
"        [6.87919568e-01, 4.10043111e-02, 2.71076121e-01],\n",
"        [4.87893811e-01, 1.47991875e-01, 3.64114313e-01],\n",
"        [1.08566107e-01, 4.51234241e-01, 4.40199652e-01],\n",
"        [1.39304789e-01, 4.50463890e-01, 4.10231321e-01],\n",
"        [1.08566107e-01, 4.51234241e-01, 4.40199652e-01],\n",
"        [3.59739437e-01, 7.59318638e-02, 5.64328699e-01],\n",
"        [2.36649455e-01, 3.12636580e-01, 4.50713965e-01],\n",
"        [2.39087068e-01, 3.70787290e-01, 3.90125642e-01],\n",
"        [6.15879037e-01, 5.80376383e-02, 3.26083325e-01],\n",
"        [3.66463739e-02, 6.15255361e-01, 3.48098265e-01],\n",
"        [3.14456990e-02, 6.50308945e-01, 3.18245356e-01],\n",
"        [1.18914637e-01, 3.76005508e-01, 5.05079855e-01],\n",
"        [8.31403880e-02, 3.87975336e-01, 5.28884276e-01],\n",
```

```
     "    mid risk       0.79      0.79      0.79       101\n",
     "\n",
     "    accuracy                           0.84       305\n",
     "   macro avg       0.84      0.85      0.84       305\n",
     "weighted avg       0.84      0.84      0.84       305\n",
     "\n"
    ]
   }
  ],
  "source": [
   "print(classification_report(y_test,pred))"
  ]
 },
 {
  "cell_type": "code",
  "execution_count": 45,
  "id": "eac2d381",
  "metadata": {},
  "outputs": [],
  "source": [
   "from sklearn.preprocessing import StandardScaler    \n",
   "scale = StandardScaler()  \n",
   "x_train= scale.fit_transform(x_train)    \n",
```

```
   "source": [
    "RF2.score(x_test,y_test)"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 91,
   "id": "fce87c6f",
   "metadata": {},
   "outputs": [
    {
     "name": "stdout",
     "output_type": "stream",
     "text": [
      "              precision    recall  f1-score   support\n",
      "\n",
      "   high risk       0.87      0.93      0.90        82\n",
      "    low risk       0.89      0.89      0.89       122\n",
      "    mid risk       0.88      0.84      0.86       101\n",
      "\n",
      "    accuracy                           0.88       305\n",
      "   macro avg       0.88      0.88      0.88       305\n",
      "weighted avg       0.88      0.88      0.88       305\n",
```

GitHub & Project Demo Link

GitHub Link

https://github.com/naanmudhalvan-SI/PBL-NT-GP--1431-1680514050/blob/ed5602c05f93480ca7604773a1bb2fa895620c62/Main%20Project/project%20.ipynb

Project Demo Link

https://drive.google.com/file/d/1WF5yVspCHu55aYwGGNxlqFoF-7DCkKBJ/view?usp=share_link