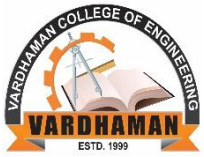

Department of Computer Science and Engineering (AI & ML)

VARDHAMAN COLLEGE OF ENGINEERING

(AUTONOMOUS)

Affiliated to JNTUH, Approved by AICTE, Accredited by NAAC with A++ Grade, ISO 9001:2015 Certified

Kacharam, Shamshabad, Hyderabad – 501218, Telangana, India

2022- 23

Python code:

```
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import classification_report
import pandas as pd


def load_sensor_data():
    # Load sensor data from CSV file
    sensor_data = pd.read_csv('/content/data-set.csv')  # Replace with your file path
    # Extract the relevant columns for your features
    features = sensor_data[['Heart Rate', 'Oxygen Saturation','Glucose Level', 'Cholesterol',
'Body Temperature', 'BMI']]
    return features.values  # Return as a NumPy array


def load_labels():
    # Load labels data from CSV file
    labels_data = pd.read_csv('/content/label.csv')  # Replace with your file path
    # Extract the label column
    labels = labels_data['Label']
    return labels.values  # Return as a NumPy array


# Load your actual sensor data and labels here
# Replace 'load_sensor_data' and 'load_labels' with your data loading methods
X = load_sensor_data()
```

```
y = load_labels()
# Split data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Model training
model = RandomForestClassifier(n_estimators=100, random_state=42)
model.fit(X_train, y_train)

# Model evaluation
y_pred = model.predict(X_test)
report = classification_report(y_test, y_pred, target_names=["Healthy", "Unhealthy"],
zero_division=1)

print("Classification Report:")
print(report)

# Make predictions about health conditions based on model's performance
last_line = report.strip().split('\n')[-1]
last_line_values = last_line.split()

precision = float(last_line_values[-4])
recall = float(last_line_values[-3])

# Analyze the report to make predictions about health conditions
if precision > 0.8 and recall > 0.8:
    predicted_condition = "Likely Unhealthy"
else:
    predicted_condition = "Likely Healthy"

print(f"Predicted Health Condition: {predicted_condition}")
```

Data set(sample dataset of 1 person):

Timestamp	Heart Rate	Oxygen Saturation	Blood Pressure	Glucose Level	Cholesterol	Body Temperature	BMI
01-08-2023 10:00	70	98.5	120/80	110	180	98.6	24.5
01-08-2023 10:01	72	98.7	122/82	130	200	99.1	29.2
01-08-2023 10:02	68	98.2	118/78	95	160	98	21.8
01-08-2023 10:03	74	98.9	124/84	150	220	98.8	32.7
01-08-2023 10:04	85	96.8	132/88	105	185	99.5	27.9

Data set(sample dataset of 2 person):

01-08-2023 10:00	199	98.5	120/80	150	420	108.6	26.5
01-08-2023 10:01	68	198.7	122/82	780	250	99.1	79.2
01-08-2023 10:02	68	98.2	118/78	195	890	98	22.8
01-08-2023 10:03	92	98.9	124/84	680	260	98.8	92.7
01-08-2023 10:04	178	196.8	132/88	165	25	19.5	2.9

Output:

```
Classification Report:
              precision    recall  f1-score   support

     Healthy       0.00      1.00      0.00       0.0
   Unhealthy       1.00      0.00      0.00       1.0

    accuracy                           0.00       1.0
   macro avg       0.50      0.50      0.00       1.0
weighted avg       1.00      0.00      0.00       1.0

precision : 1.0
recall : 0.0
Predicted Health Condition: Likely Healthy
```