Design and Implementation of Pulse oximeter to monitor and predict Patient's health

Dharmendrasinh J. Revar[a], Hemal D. Nayak[b],D.K Jhala[c]

[a] PG Student,[b] Assistant Professor,Department of mobile communication and network technology,GTU- Graduate school of engineering and Technology,Gandhinagar, Gujarat,India

[C] Visiting Faculty,BISAG - Bhaskaracharya Institute for Space Applications and Geo-informatics,Gandhinagar,Gujarat,India

ABSTRACT:

Pulse oximeters are widely used as a non-intrusive way to measure blood-oxygen saturation. The aim of this design is to examine the accurate oxygen saturation level and to predict the patients' health condition using machine learning. Generally the Spo2 level of normal people is 95% to 100%.The Pulse oximeter will able to take the reading of Spo2 and heart rate by using max30100 module. The data of patients is monitor using this design and will be stored in the cloud using IoT technology(Node mcu Wi-Fi module).The Patients health will be predict by machine learning algorithm in cloud. The live data of patients will facilated to the patients/doctor by using mobile app. Using this historical Data the ML(Machine learning) algorithm predict the patient's health into 9 health categories are Normal, Tachycardia, Bradycardia, somehow reduced Spo2,Bradycardia and middle Spo2,Tachycardia and middle Spo2,Spo2 critical, COPD and critical condition.

KEYWORDS:

Pulse oximeter, Node mcu, MAX30100, oxygen level, Machine learning, IoT

1. Introduction

1.1 Introduction to pulse oximeter

The pulse oximeter is the device that works on the principle of the pulse oximetry. The pulse oximetry is the method for monitoring the people's oxygen saturation (SpO2) without harming their bodies. The Standard SpO2 level is between 95 to 100 percent. SpO2 level is mainly depends on R (ratio) value. R value is the ratio value of Red light to Infrared light. The wavelength of Red is at 660nm and IR (infrared) is at 940nm.

The pulse oximeter checks to see if people's blood oxygen level is within a healthy range or not. When your oxygen level is down from normal range, it is called hypoxemia.

When person's oxygen level falls down from normal range, the person's lungs starting to become narrow, it is res started to slower down the blood flow by the lungs. When this things happen more stress has been placed on the heart. This could be lead to Serious Heart failure problem for peoples.

Some symptoms of hypoxemia is given below:

- Feeling stress on Heart
- Problem in normal breathing
- Feeling stress
- Feeling restlessness

1.2 Introduction to Internet of Things (IoT):

IoT (Internet of Things) is a technology that can connect all the devices to the cloud and devices are controlled and monitor through the cloud. After IoT is coming into the Picture the sensor are attached with device that can give the values that can be measured from any were in the world through the cloud by accessing remotely. The IoT has been used in many sectors like smart home, smart city, smart hospitals, smart Television and Smart Phones etc. IoT has reduced the cost of monitoring the patients through cheap sensors that can be helpful for doctor as well as patients. The use of IoT devices has been increased rapidly day by day. IoT can change the way of our Living.

1.3 Introduction to Machine Learning (ML):

Machine leaning is the way to teach the Algorithms to learn from its past experience And updated itself .Machine learning is a part of AI(artificial intelligence).Machine Learning algorithms are build model in Joblib file on sample data. On the sample Data the partition should be possible for train and test data.so that it is used later to measure the accuracy if the algorithms. There are many software used for performing machine learning algorithms like spyder, pycharm, Jupyter notebook etc. There are three types of Machine Learning given below:

- Supervised machine learning
- Unsupervised machine learning
- Reinforcement machine learning

In supervised Machine learning Labels are attached with data. In unsupervised Machine learning labels are not attached with data.in unsupervised machine learning Clustering-based approach is taken. In Reinforcement Machine learning feedback Based approach is taken. Based on the Data the machine learning algorithms are two types given below:

- Classification
- Regression

The programmer is decide to base on data they want to choose classification or regression. Normally when we want to classify some values at that time Classification algorithms are used and when we want to predict continues values like temperature at that time regression algorithms are used.

2. Implementation of Pulse oximeter

2.1 Components needed:

- JHD 162A 16*2 LCD display
- ESP8266(NODEMCU) Wi-Fi module
- Bread board
- Male to Male, Female to Male, And Female To Female connecter
- MAX 30100 module
- I2C Module

2.2 Interfacing LCD, Node MCU and MAX30100:

The software for writing the code is Arduino IDE (Integrated Development Environment).The code is similar to C language code. The sketch written in Arduino IDE when it complied successfully after that code is uploaded to Node MCU. The compiling code is given in Fig.3.5.The results of Spo2 and Heart rate shown in display on Fig.3.6.

For Compilation of code the following libraries are added compulsory:

- MAX30100_PulseOximeter.h
- Wire.h
- ESP8266WiFi.h
- LiquidCrystal_I2C.h

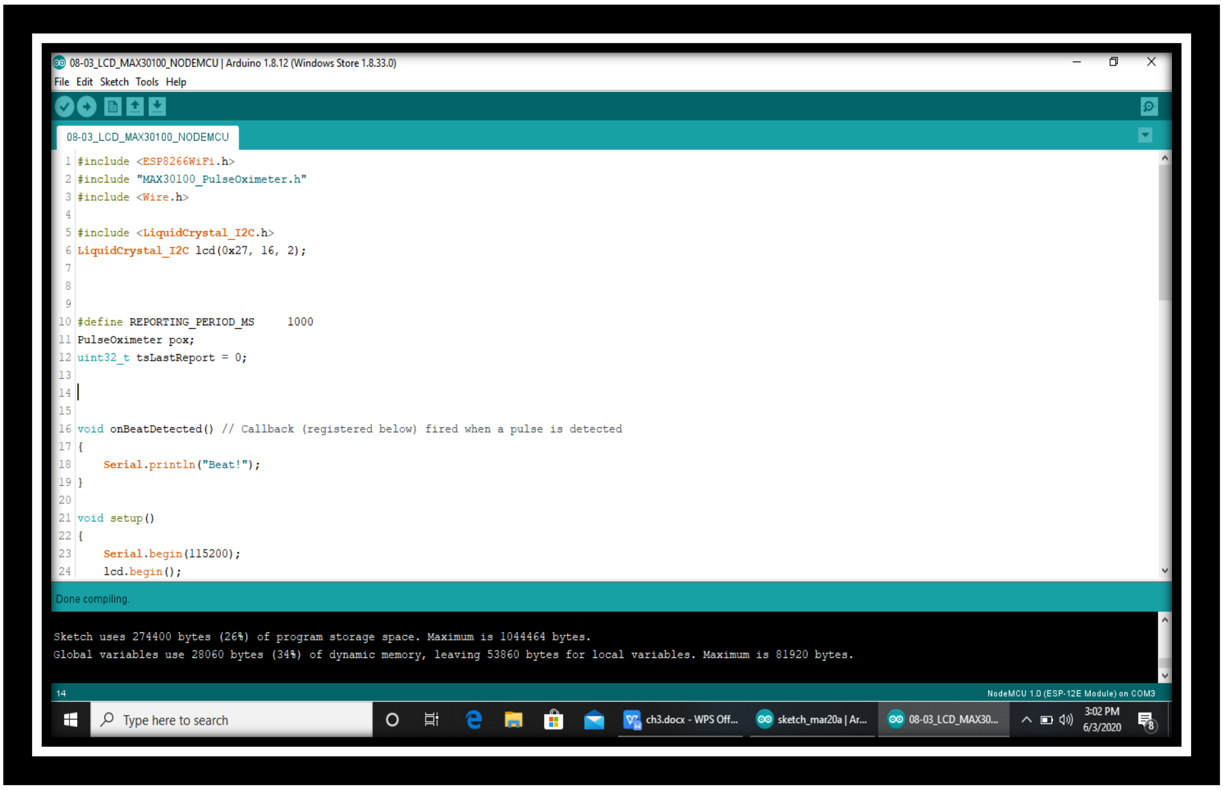


Fig 2.1: Code compilation of LCD, Node MCU and MAX30100

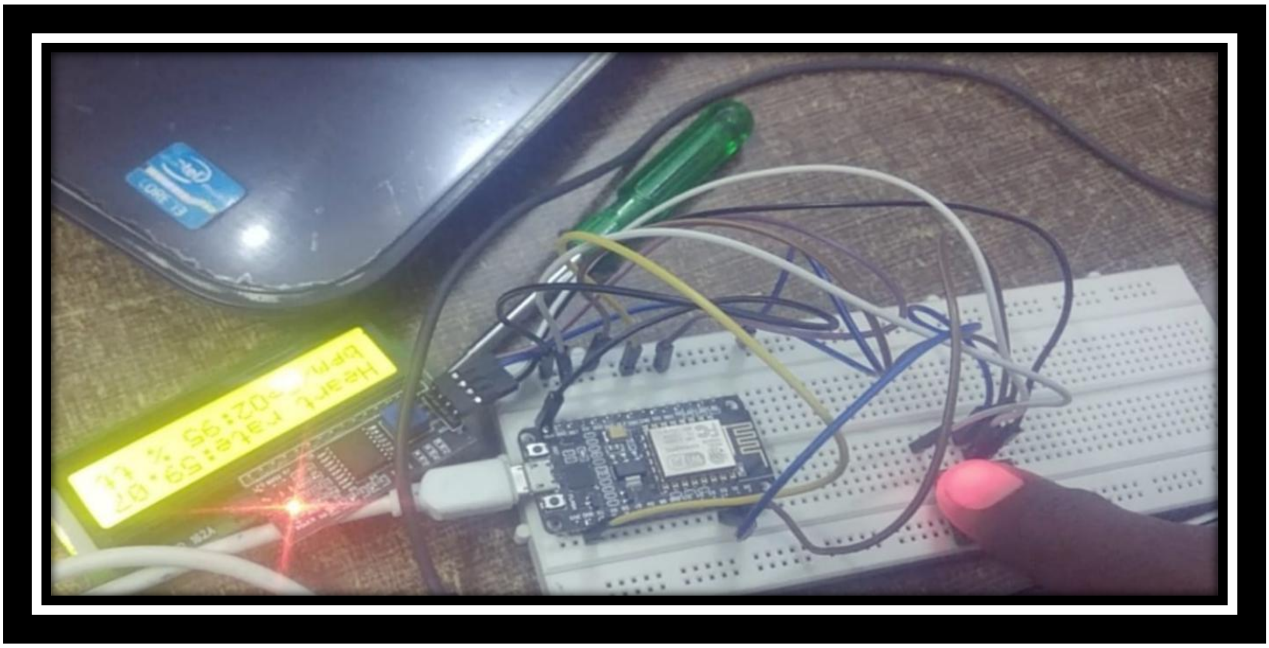


Fig 2.2: Result of Spo2 and Heart rate on Display

3. Preparation of App on Blynk platform:

Blynk is open source IoT platform that provide connectivity to connect the IoT devices to them. Blynk is also provide the facility to making our own app as per our device configuration.

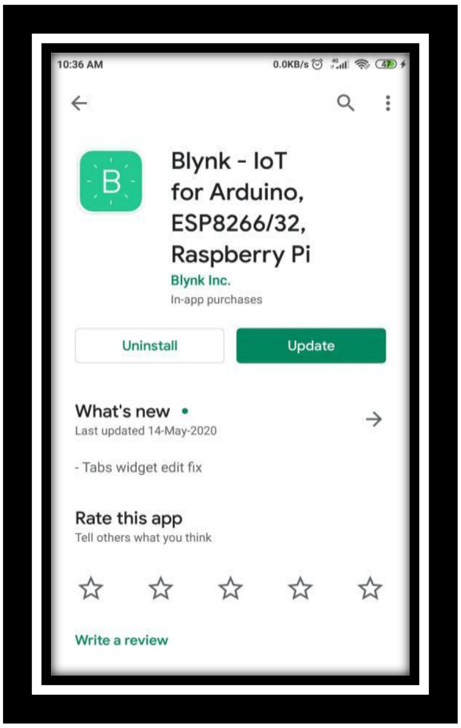


Fig 3.1: Blynk app Platform

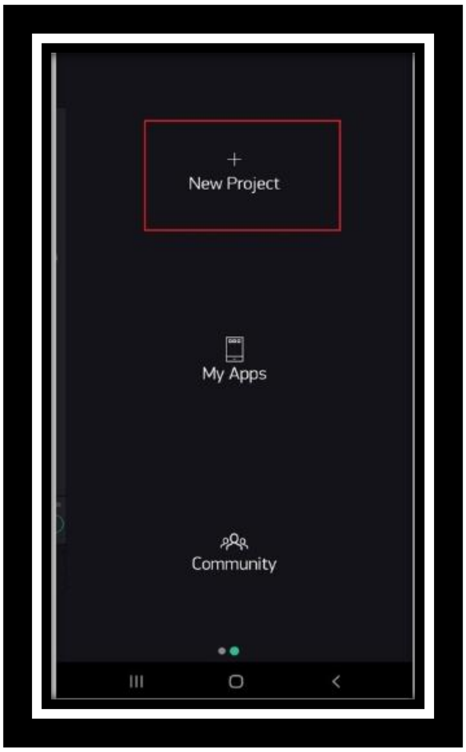


Fig 3.2: Creating a New project

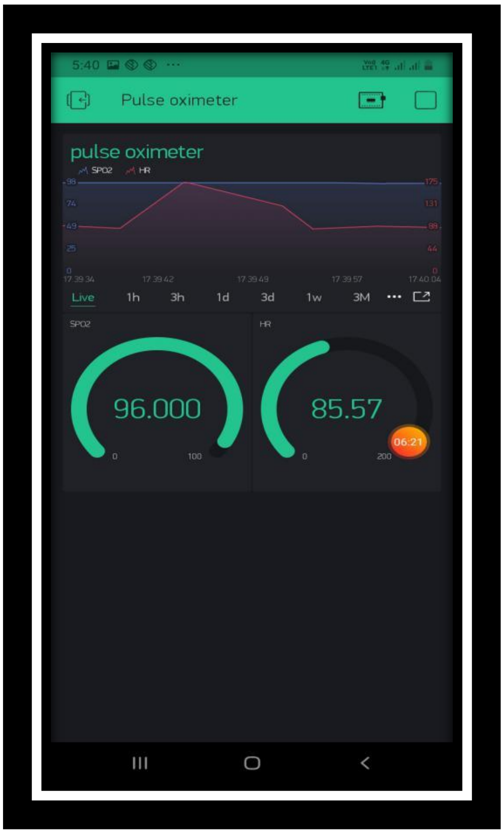


Fig 3.3: Pulse oximeter app on Blynk Platform

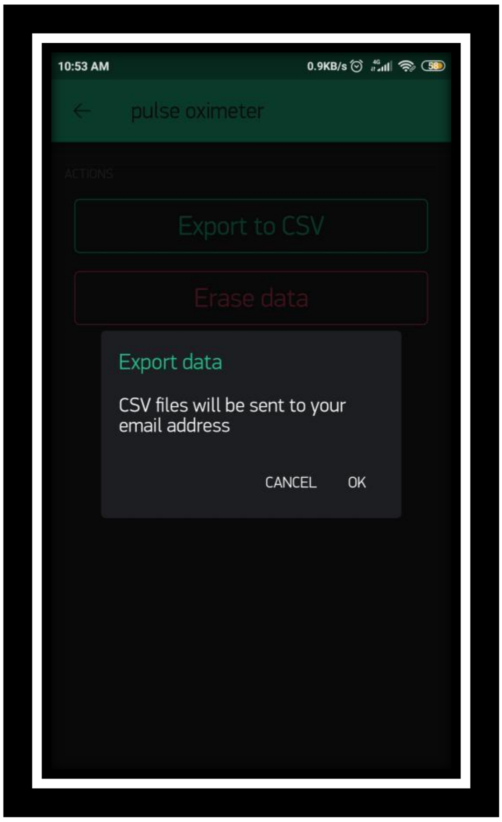


Fig 3.4: historical data sent to mail

Fig 3.5: Code compilation on Arduino IDE for Blynk Platform

Fig 3.1 shows Blynk Platform on Google Play store. Fig 3.2 shows the way to create new project on Blynk Platform. The data of Spo2 was stored in V5 (virtual Pin) and The Data of Heart rate (HR) was stored in V6 (Virtual pin).Fig 3.3 shows the application of Pulse oximeter on Blynk Platform. The real time data of Spo2 and HR is shows in label and Historical data is shows in graphical form. The historical Data is stored in Blynk cloud. Fig 3.4 shows the historical data is sent in E-Mail for Further analysis.Fig3.5 shows the compilation is done on Arduino IDE software.

4. Creating Health conditions on Historical data and data validation by doctor

4.1 health conditions:

The health condition are create on 1304 samples of data shown in Table 4.1

Health conditions	Labels	Samples	conditions
Some how reduced	1	173	SPO_2 : 92 to 95
Bradycardia and middle SPO_2	2	18	HR:<60 SPO_2 : 92 to 95
Tachycardia and middle SPO_2	3	59	HR:>100 SPO_2:92 to 95
COPD[3]	4	32	SPO_2:88 to 91
Critical Condition	5	46	SPO_2< 91 HR:<60 and >100
Bradycardia[1]	6	71	HR:<60
Tachycardia[2]	7	179	HR:>100
SPO_2 critical[5]	8	21	SPO_2< 87
Normal[6]	9	704	SPO_2 : 96 to 100 HR:60 to 100

Table 4.1: Health conditions

Tachycardia:

Tachycardia happens when person’s heart beat more Than 100 time per minute. There are three types of Tachycardia.

- Supraventricular
- Ventricular
- Sinus tachycardia

The main Reasons for Tachycardia are strenuous exercise, a fever, fear, stress, anxiety, certain medications, street drugs and smoking.

Bradycardia:

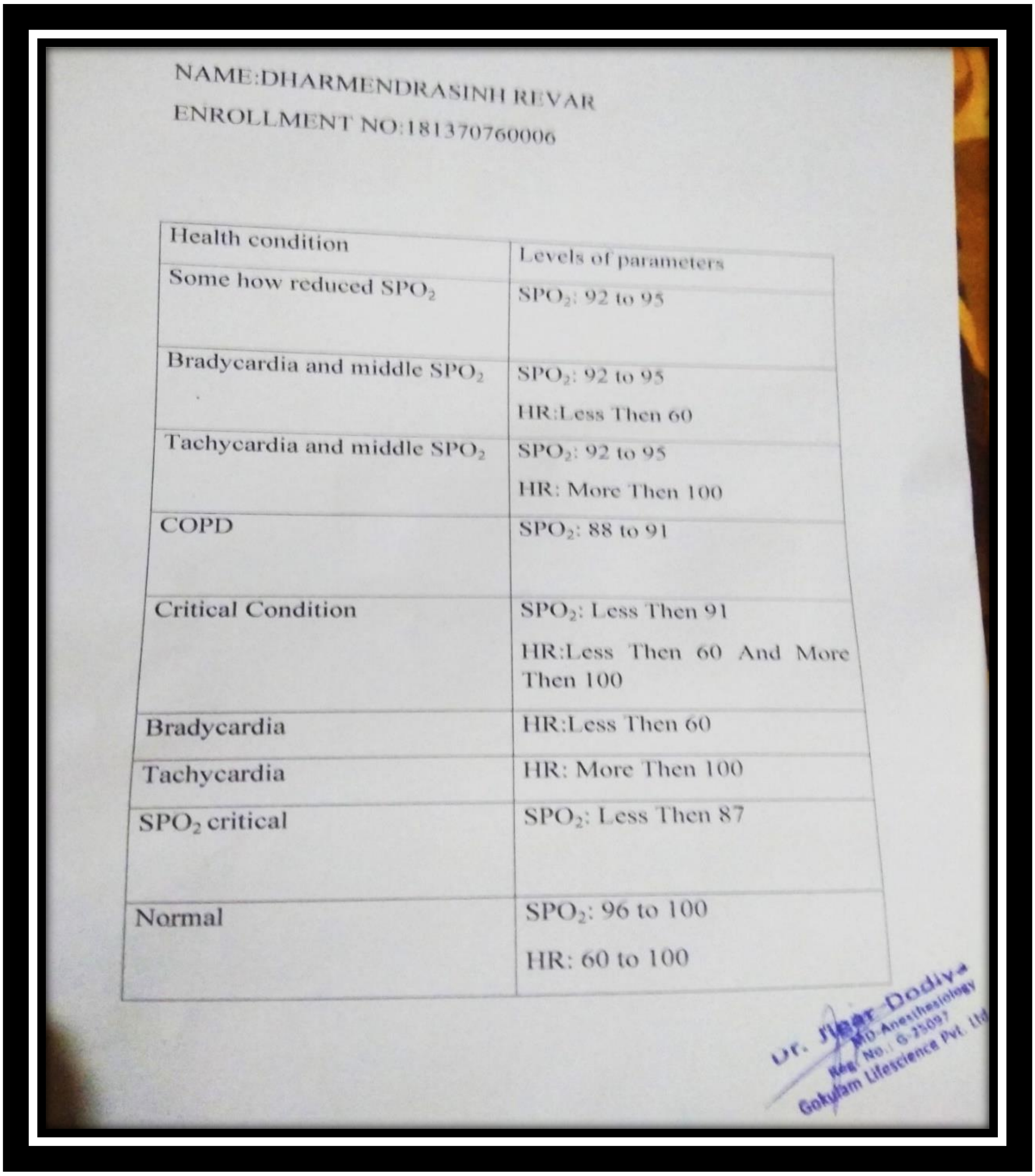
Bradycardia happen when person’s heart beat slower than 60 Time per minute. Normal heart beat range is Between 60 to 100. The symptoms of Bradycardia are Fatigue, shortness of breath, Pain in chest and memory problem. The main risk factor for Bradycardia is age. Bradycardia normally happens in older age Persons.

Chronic obstructive pulmonary disease (COPD):

COPD is common lung issue. COPD people normally have breathing problem and smoking is the main reason for COPD. But some people do not smoke still COPD Happens. Heavy amount of pollution also cause COPD.

4.2 Data Validation by Doctor:

Data validation is done by Dr.jigar dodiya (Anesthesiology) From Gokul super specialty, Rajkot as shown in Fig 4.1



NAME:DHARMENDRASINH REVAR
ENROLLMENT NO:181370760006

Health condition	Levels of parameters
Some how reduced SPO_2	SPO_2: 92 to 95
Bradycardia and middle SPO_2	SPO_2: 92 to 95 HR:Less Then 60
Tachycardia and middle SPO_2	SPO_2: 92 to 95 HR: More Then 100
COPD	SPO_2: 88 to 91
Critical Condition	SPO_2: Less Then 91 HR:Less Then 60 And More Then 100
Bradycardia	HR:Less Then 60
Tachycardia	HR: More Then 100
SPO_2 critical	SPO_2: Less Then 87
Normal	SPO_2: 96 to 100 HR: 60 to 100

Dr. Jigar Dodiya
MD-Anesthesiology
Reg. No.: G-25097
Gokulam Lifescience Pvt. Ltd

Fig 4.1 Data validation done by doctor

5. Machine learning algorithms of validate data

5.1 Data analysis:

Fig 5.1 shows statistics of Data set of 1304 samples like number of variables, number of observations, missing cells, duplicate rows, total size of memory, average size of memory etc.

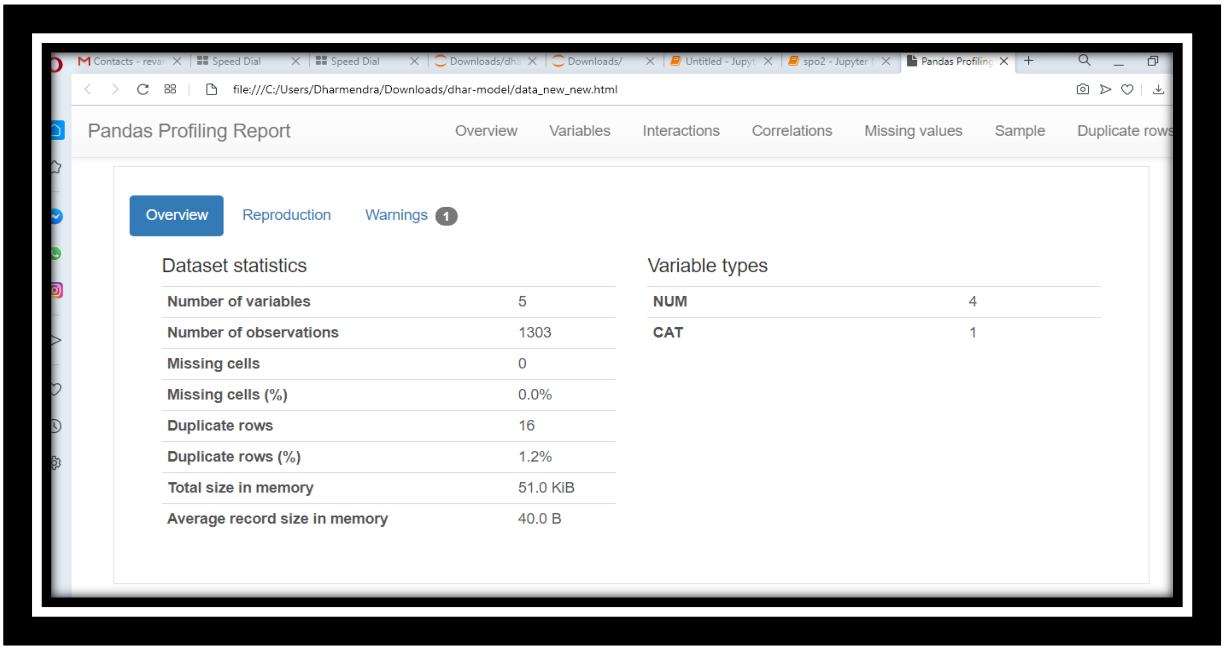


Fig 5.1: Data set statistics

Fig 5.2 shows details about Spo2 like mean, maximum, zeros, memory Size, minimum, zeros, unique, infinite, missing etc. the graph shows that Spo2 values on X-axis and number of samples on Y-axis.

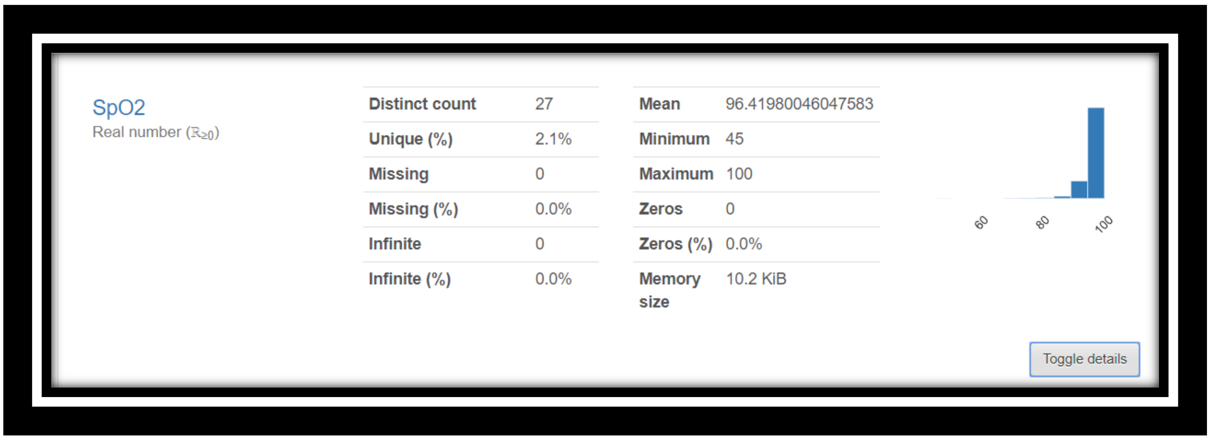
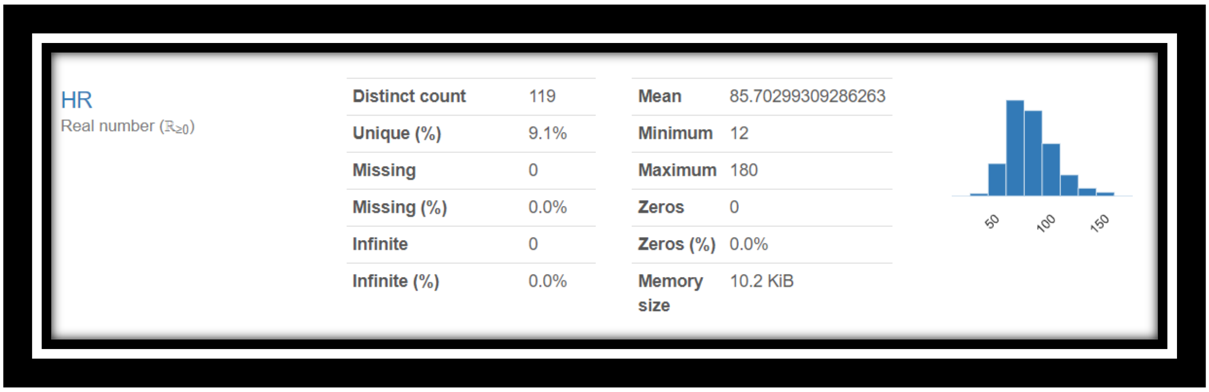


Fig 5.2:Spo2 details

Fig 5.3 shows details about Spo2 like mean, maximum, zeros, memory Size, minimum, zeros, unique, infinite, missing etc. the graph shows that Spo2 values on X-axis and number of samples on Y-axis.

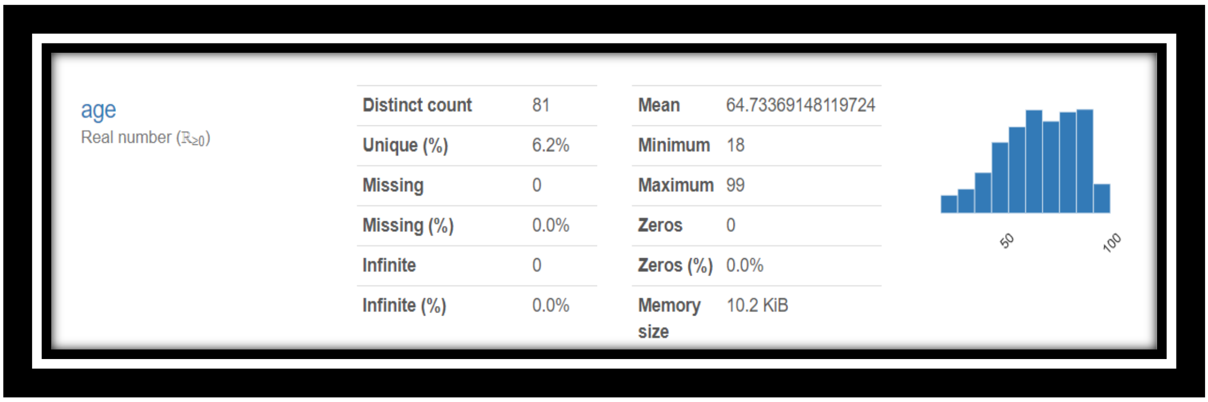


HR
Real number ($\mathbb{R}_{\geq 0}$)

Distinct count	119	Mean	85.70299309286263
Unique (%)	9.1%	Minimum	12
Missing	0	Maximum	180
Missing (%)	0.0%	Zeros	0
Infinite	0	Zeros (%)	0.0%
Infinite (%)	0.0%	Memory size	10.2 KiB

Fig 5.3: HR details

Fig 5.4 shows details about HR like mean, maximum, zeros, memory Size, minimum, zeros, unique, infinite, missing etc. the graph shows that age values on X-axis and number of samples on Y-axis.

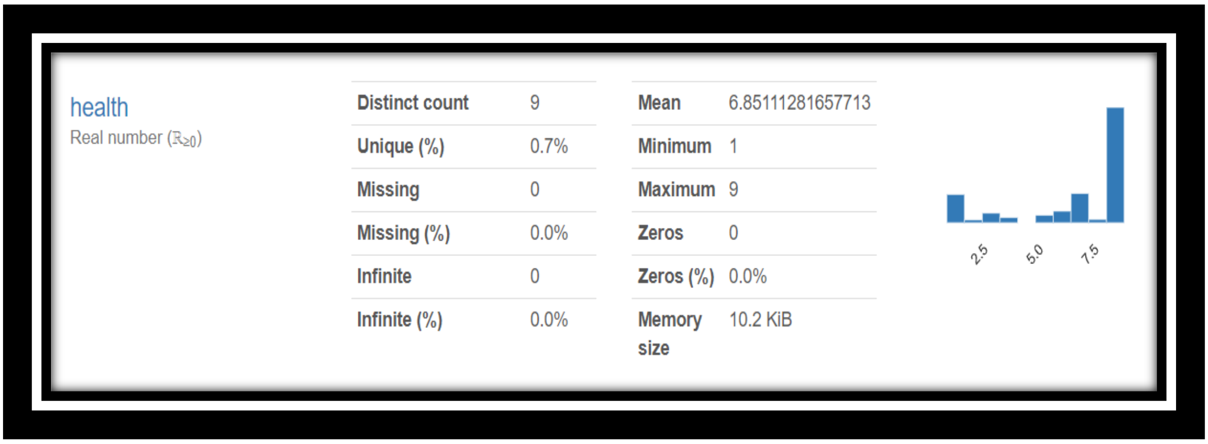


age
Real number ($\mathbb{R}_{\geq 0}$)

Distinct count	81	Mean	64.73369148119724
Unique (%)	6.2%	Minimum	18
Missing	0	Maximum	99
Missing (%)	0.0%	Zeros	0
Infinite	0	Zeros (%)	0.0%
Infinite (%)	0.0%	Memory size	10.2 KiB

Fig 5.4: age details

Fig 6.18 shows details about health like mean, maximum, zeros, memory Size, minimum, zeros, unique, infinite, missing etc. the graph shows that HR values on X-axis and number of samples on Y-axis.



health
Real number ($\mathbb{R}_{\geq 0}$)

Distinct count	9	Mean	6.85111281657713
Unique (%)	0.7%	Minimum	1
Missing	0	Maximum	9
Missing (%)	0.0%	Zeros	0
Infinite	0	Zeros (%)	0.0%
Infinite (%)	0.0%	Memory size	10.2 KiB

Fig 5.5: health details

The Figures are given above are from the report of EDA (exploratory data analysis) from the validate data set of 1304 samples. It is also called Pandas profiling Report.

5.2 Algorithms Comparison:

K nearest neighbor (KNN):

KNN is also supervised learning algorithm. KNN is used for classification purpose. KNN is also known as lazy algorithm because it is only summarized learning process and do not learn or take its own decision itself. KNN basically identifies the data points that are separated into various classes to predict the classification of a given sample point. This predictions or classifications are done based on similarity measure. The similarity measure means KNN finds its nearest points based on Euclidean distance. The number of nearest points to find are defined by the value of K. suppose defined K value is 5, then KNN finds five nearest sample points from the test point and checks the labels of that five nearest points. KNN predicts or gives the classification output for the test sample based on the majority class of labels. For example, for K = 3 and for binary classification, two labels are of same class and third is of another class. Then KNN predicts the majority class label as output class. This can be understood by showing figure below.

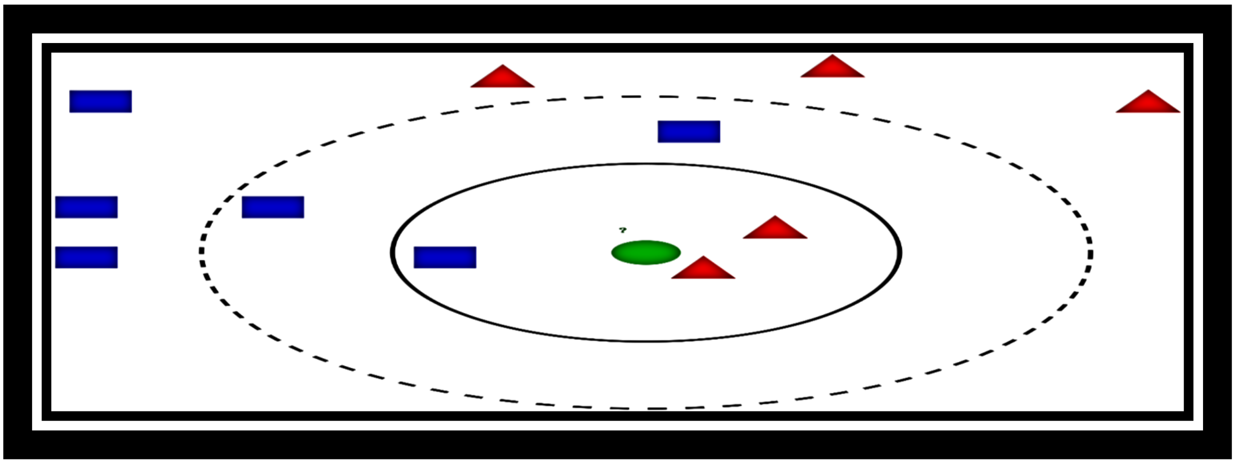


Fig 5.6: KNN Algorithm

Sample point 

Class A 

Class B 

```
In [23]: # The baseline predictions are the historical averages
         baseline_preds = test_features[:, feature_list.index('age')]
         # Baseline errors, and display average baseline error
         baseline_errors = abs(baseline_preds - test_labels)
         print('Average baseline error: ', round(np.mean(baseline_errors), 2))

         Average baseline error:  57.74
```

Fig 5.7: Average baseline error of KNN

```
In [34]: from sklearn.metrics import mean_squared_error
         mse = mean_squared_error(test_labels, predictions)
         rmse = np.sqrt(mse)

In [35]: rmse

Out[35]: 2.7464422049764807

In [36]: #testset = [[test_features]]
         #predictions = [[test_labels]]
         #accuracy = getAccuracy(testset, predictions)
         #print(accuracy)

In [37]: mape = 100 * (errors / test_labels)
         accuracy = 100 - np.mean(mape)
         print('Accuracy:', round(accuracy, 2),'%.')

         Accuracy: 18.46 %.
```

Fig 5.8: RMSE and Accuracy of KNN

```
In [25]: # we imported the model using sci-learn kit

         # Use the forest's predict method on the test data
         predictions = rf.predict(test_features)
         # Calculate the absolute errors
         errors = abs(predictions - test_labels)
         # Print out the mean absolute error (mae)
         print('Mean Absolute Error:', round(np.mean(errors), 2), '')
         print(errors)
         #print(n12)

         Mean Absolute Error: 1.12
         [0 0 0 0 3 0 0 0 0 0 0 0 8 0 0 0 0 0 0 0 0 3 0 0 7 0 0 0 0 1 0 8 0 0 0 0 1
          8 8 7 0 0 0 0 0 8 0 0 8 0 4 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 8 0 0 0 0 0 0 0
          0 0 8 0 0 8 0 0 0 0 0 0 0 8 8 1 8 0 0 0 8 0 0 0 0 6 0 0 0 8 0 0 2 0 0 0 0
          0 0 0 8 8 0 0 4 0 0 0 0 0 4 0 0 0 0 0 0 0 2 0 0 0 0 0 0 0 0 0 0 8 0 0 0 0
          0 0 0 8 0 0 0 4 0 0 0 0 0 0 0 0 0 0 0 0 3 2 0 4 4 0 4 8 0 0 0 0 8 0 4 0 0
          4 0 0 0 0 0 0 0 0 0 4 3 0 8 0 0 0 4 8 4 0 0 0 0 0 2 0 0 0 0 0 8 0 0 0 0 0
          0 0 0 0 0 0 0 0 0 0 0 0 0 4 4 0 0 0 0 0 0 0 0 0 4 0 0 0 0 1 0 0 0 8 0 0 0
          0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 4 0 8 0 0 0 0 0 0 4 0 0 0 0 0 0 8
          0 8 8 0 0 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 8 0 8 0 2 0 8 0]
```

Fig 5.9: MAE of KNN

Decision Tree (DT):

Decision tree algorithm is same as tree type structure. Decision tree algorithm is used for both the classification as well as regression problem. Decision tree algorithm mostly depends on the independent variables. Independent variables are taken as master node. This master node will have conditions over the features of the dataset. This master node has condition through which it decides on which leaf node to navigate next. Once the last leaf node is reached after satisfying all the conditions, the output is predicted. The better the sequence of conditions the better the output of algorithm. The term Information Gain is used to select the conditions in nodes. To derive the tree structure of the algorithm, a greedy, recursive based algorithm is used.as shown in FIG.6.31 the Decision will be taken based on Predicted score.

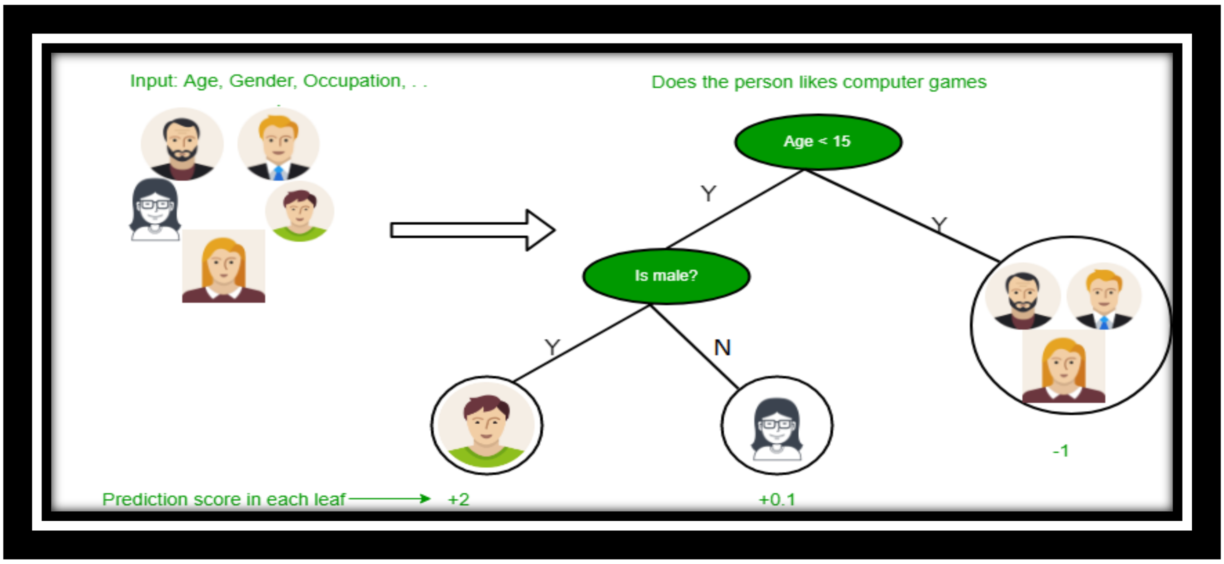


Fig 5.10: Decision Tree Algorithm

```
In [23]: # The baseline predictions are the historical averages
         baseline_preds = test_features[:, feature_list.index('age')]
         # Baseline errors, and display average baseline error
         baseline_errors = abs(baseline_preds - test_labels)
         print('Average baseline error: ', round(np.mean(baseline_errors), 2))

         Average baseline error:  57.74
```

Fig 5.11: Average baseline error of Decision Tree

```
In [34]: from sklearn.metrics import mean_squared_error
         mse = mean_squared_error(test_labels, predictions)
         rmse = np.sqrt(mse)

In [35]: rmse

Out[35]: 0.0

In [36]: #testset = [[test_features]]
         #predictions = [[test_labels]]
         #accuracy = getAccuracy(testset, predictions)
         #print(accuracy)

In [37]: mape = 100 * (errors / test_labels)
         accuracy = 100 - np.mean(mape)
         print('Accuracy:', round(accuracy, 2),'%.')

         Accuracy: 100.0 %.
```

Fig 5.12: RMSE and Accuracy of Decision Tree

```
In [25]: # we imported the model using sci-learn kit

         # Use the forest's predict method on the test data
         predictions = rf.predict(test_features)
         # Calculate the absolute errors
         errors = abs(predictions - test_labels)
         # Print out the mean absolute error (mae)
         print('Mean Absolute Error:', round(np.mean(errors), 2), '')
         print(errors)
         #print(n12)

         Mean Absolute Error: 0.0
         [0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
          0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
          0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
          0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
          0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
          0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
          0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
          0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
          0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
          0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
          0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
          0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
          0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
          0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
```

Fig 5.13: MAE of Decision Tree

Random Forest (RF):

Random Forest is the Machine learning algorithm that can be operate by constructing multiple decision trees. The final decision is made based on majority of the trees as shown in FIG 6.34. A decision tree is a tree-shaped diagram used to determine a course of action. Each branch of the tree represents a possible decision, occurrence, or reaction.[12]

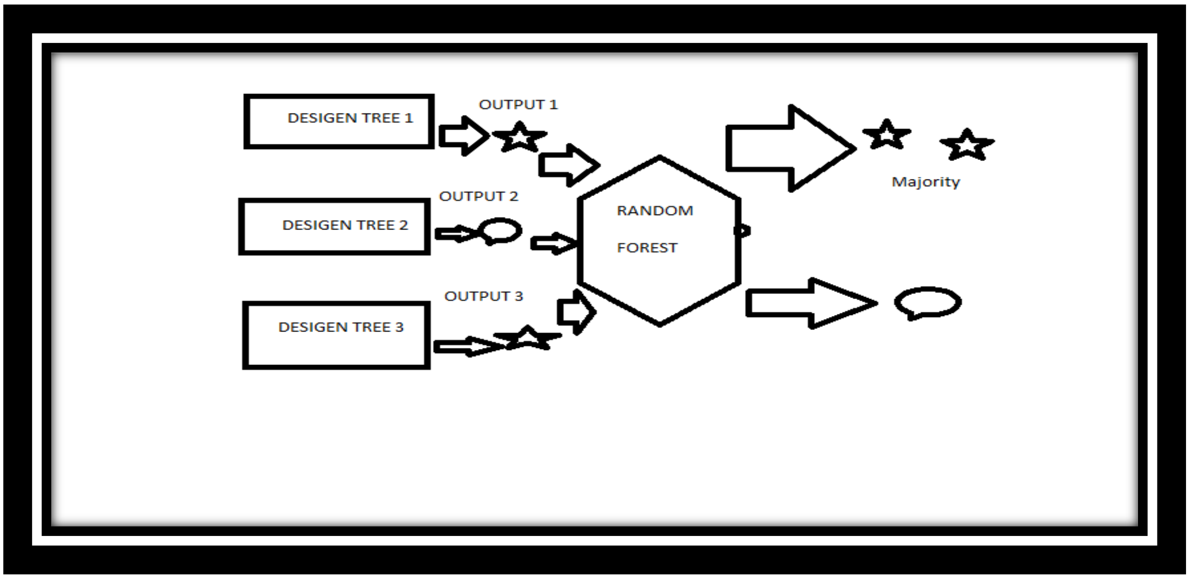


Fig 5.14: Random Forest Algorithm

```
In [23]: # The baseline predictions are the historical averages
         baseline_preds = test_features[:, feature_list.index('age')]
         # Baseline errors, and display average baseline error
         baseline_errors = abs(baseline_preds - test_labels)
         print('Average baseline error: ', round(np.mean(baseline_errors), 2))

         Average baseline error:  57.74
```

Fig 5.15: Average baseline error of Random forest

```
In [34]: from sklearn.metrics import mean_squared_error
         mse = mean_squared_error(test_labels, predictions)
         rmse = np.sqrt(mse)

In [35]: rmse

Out[35]: 0.04017416072781169

In [36]: #testset = [[test_features]]
         #predictions = [[test_labels]]
         #accuracy = getAccuracy(testset, predictions)
         #print(accuracy)

In [37]: mape = 100 * (errors / test_labels)
         accuracy = 100 - np.mean(mape)
         print('Accuracy:', round(accuracy, 2),'%.')

         Accuracy: 99.81 %.
```

Fig 5.16: RMSE and Accuracy of Random forest

```
In [25]: # we imported the model using sci-learn kit

         # Use the forest's predict method on the test data
         predictions = rf.predict(test_features)
         # Calculate the absolute errors
         errors = abs(predictions - test_labels)
         # Print out the mean absolute error (mae)
         print('Mean Absolute Error:', round(np.mean(errors), 2), '')
         print(errors)
         #print(n12)

         Mean Absolute Error: 0.01
```

Fig 5.17: MAE of Random forest

The features are in the validate data set Spo2, Gender, heart rate (HR) and age. The labels is health. The software used for performing algorithm is Jupyter Notebook. The codes are written in python language. In the validate data set 25% data was taken for testing and 75% data was taken for training. The Aim is classify the health. Three main algorithms for classification is K Nearest Neighbor, Random Forest and Decision tree. The comparison on three algorithm is shown in Table 5.1 on the parameters are MAE (mean absolute error), Average Baseline error, RMSE (root mean square error) and Accuracy.

parameters	K-nearest neighbor	Decision Tree	Random Forest
Mean absolute error	1.12	0.0	0.01
Average baseline error	57.74	57.74	57.74
Root mean square error	2.74674	0.0	0.040
Accuracy	18.46%	100%	99.81%

Table 5.1: Comparison of algorithms

6. Results and Conclusion:

Results:

41 samples are taken from normal to critical people, Total six Person's samples was taken and enter the mean values of Spo2 and HR (heart rate) of Samples in Decision Tree Algorithm. The results are shown in Table6.1.

Gender	**Age**	**Spo2**	**HR**	**Health**
1(MALE)	**26**	**96**	**77**	**Normal(9)**
1(MALE)	**37**	**89**	**78**	**COPD(4)**
2(FEMALE)	**64**	**90**	**52**	**Critical condition(5)**
2(FEMALE)	**35**	**97**	**82**	**Normal(9)**
1(MALE)	**68**	**96**	**105**	**Tachycardia(7)**
2(FEMALE)	**52**	**97**	**54**	**Bradycardia(6)**

TABLE 6.1: Results of Decision Tree Algorithm on Jupyter Notebook

Conclusion:

As Parameter shown in Comparison Table 5.1 Decision Tree Algorithm can give Better Output than KNN (K nearest Neighbor) And Random Forest Algorithms. The project is very helpful For Doctor As well as Patient for Monitoring the Condition. This project can be enhanced in future for adding more medical Parameters.

Acknowledgments: We Thank to All Staff members of BISAG and GTU_GSET For helping us and inspiring us.

References:

[1] Rodriguez LR, Kotin N, Lowenthal D, Kattan M: A study of pediatric house staff's knowledge of pulse oximetry. Pediatrics 1994;93:810-813

[2] Stoneham MD, Saville GM, Wilson IH: Knowledge about pulse oximetry among medical and nursing staff. Lancet 1994;344:1339- 1342

[3] Kelleher JF: Pulse oximetry. J Clin Monit 1989;5:37-62

[4] Grace RF: Pulse oximetry. Gold standard or false sense of security? Med J Aust 1994; 160:638-644

[5] Tremper KK, Barker S J: Pulse oximetry. Anesthesiology 1989; 70:98-108

[6] Wukitsch MW, Petterson MT, Tobler DR, Pologe JA: Pulse oximetry: analysis of theory, technology, and practice. J Clin Monit 1988;4:290-301

[7] L. Schallom, C. Sona, M. McSweeney, and J. Mazuski, “Comparison of forehead and digit oximetry in surgical/trauma patients at risk for decreased peripheral perfusion,” Heart & Lung: The Journal of Acute and Critical Care, vol. 36, no. 3, pp. 188–194, 2007.

[8] R. D. Branson and P. D. Mannheimer, “Forehead oximetry in critically ill patients: the case for a new monitoring site.,” Respiratory care clinics of North America, vol. 10, no. 3, pp. 359–67, 2004.

[9] T. Rusch, J. Scharf, and R. Sankar, “Alternate pulse oximetry algorithms for spo/sub 2/computation,” in Engineering in Medicine and Biology Society, 1994. Engineering Advances: New Opportunities for Biomedical Engineers. Proceedings of the 16th Annual International Conference of the IEEE, vol. 2, pp. 848–849, IEEE, 1994.

[10] J. Cho, Y. Sung, K. Shin, D. Jung, Y. Kim, and N. Kim, “A preliminary study on photoplethysmogram (ppg) signal analysis for reduction of motion artifact in frequency domain,” in Biomedical Engineering and Sciences (IECBES), 2012 IEEE EMBS Conference on, pp. 28–33, IEEE, 2012.