**PROJECT SYNOPSIS ON: Heart Attack Prediction**

***Midterm Report Submitted in Partial Fulfillment of the Requirements***

***for the degree of***

**Bachelor of Technology in Applied Electronics and Instrumentation Engineering**

**By**

**Md Irshad**

**Nishant Shrivastava**

**Mithlesh Yadav**

**Satyam Anand**

**Vishnu Kumar**

**13005317046**

**13005317042**

**13005317045**

**13005317026**

**13005317006**

**Under the guidance of**

**Subhajit Bhowmick**

**Techno Main, Salt Lake**

**EM 4/ Salt lake City, Sector V**

**Kolkata – 700091**

**2021**

**Abstract:**

Our main focus in this project is to build an IoT device using that we can predict a person risk of heart attack. Here we have used Machine Learning Algorithm (Support Vector Machine) which is a classification algorithm for prediction. Basically the heart our project is Python Machine learning model and using Web based programming we will perform data manipulation from the cloud and serve it in the client side using latest technologies and the hardware part will be running on the WiFi Based Arduino setup with a sensor which measures the heart rate of the person.

By performing Data Analysis and Data Analytics we have found some crucial variable for our effective prediction

These variables are very important variables which will be user inputs. They will be feed with our IoT sensor data and then our model will predict.

In total for the purpose of classification we have choose 5 variables.

1. Sex (enter by user)

2. Age (enter by user)

3. Cholesterol (Programmatically added data)

4. Blood pressure (Programmatically added data)

5. Heart Rate (Sensor data)

Due to some limitations like access to more hardware and software packages we will not be able to measure the values of Cholesterol and Blood pressure that’s why we will be trying a programmatic way to solve this problem.

We will be hard coding some approx values according to the age of users.

We choose SVM classifier as our preferred algorithms because our datasets is small if we get some more data then we will be using deep neural networks.

There are three layer of this project

I. Python ML Model (SVM classifier)

2. IoT device

3. Connecting Server and Client. (Example: Web Application)

**Table of contents/Index:**

* Introduction
* Detailing of the project work and simulation
* Result Set
* Discussion
* Remaining Work
* References and Bibliography

**Introduction:**

Heart diseases are the main reasons for death worldwide. According to the survey of the World Health Organization (WHO), 17.5 million total global deaths occur because of heart attacks and strokes. More than 75% of deaths from cardiovascular diseases (CVD) occur mostly in middle-income and low-income countries.

Also, 80% of the deaths that occur due to CVDs are because of stroke and heart attack. Therefore, detection of cardiac abnormalities at the early stage and tools for the prediction of heart diseases can save a lot of life and help doctors to design an effective treatment plan which ultimately reduces the mortality rate due to cardiovascular diseases. Due to the development of advance healthcare systems, lots of patient data are nowadays available at open source which can be used for designing predictive models for Cardiovascular diseases. Machine learning is a discovery method for analyzing big data from an assorted perspective and encapsulating it into

useful information.

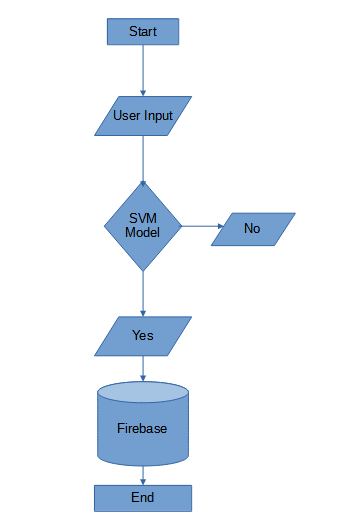
Nowadays, a huge amount of data pertaining to disease diagnosis, patients etc. are generated by healthcare industries. Data mining provides a number of techniques which discover hidden patterns or similarities from data. Therefore, in this paper, a machine learning algorithm is proposed for the implementation of a heart disease prediction system which was validated on two open access heart disease prediction datasets.

The sensors collect the data after a specific time, analyze it and use it to initiate the required action, and provide an intelligent cloud-based network for analysis, planning and decision making.

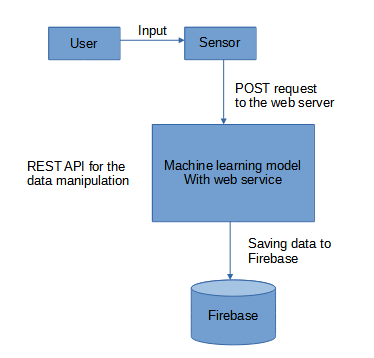
**Detailing of the project work and simulation:**

The heart of our project lies in Machine learning and in Visualization of user’s data.

This project needs user’s data to work or predict the outcome of heart attack whether he/she has a risk of heart attack or not.

**Machine learning Flow Chart**

The incoming request made by the user using Arduino is received by Machine learning API and then the output data will be send to database and stored in it.



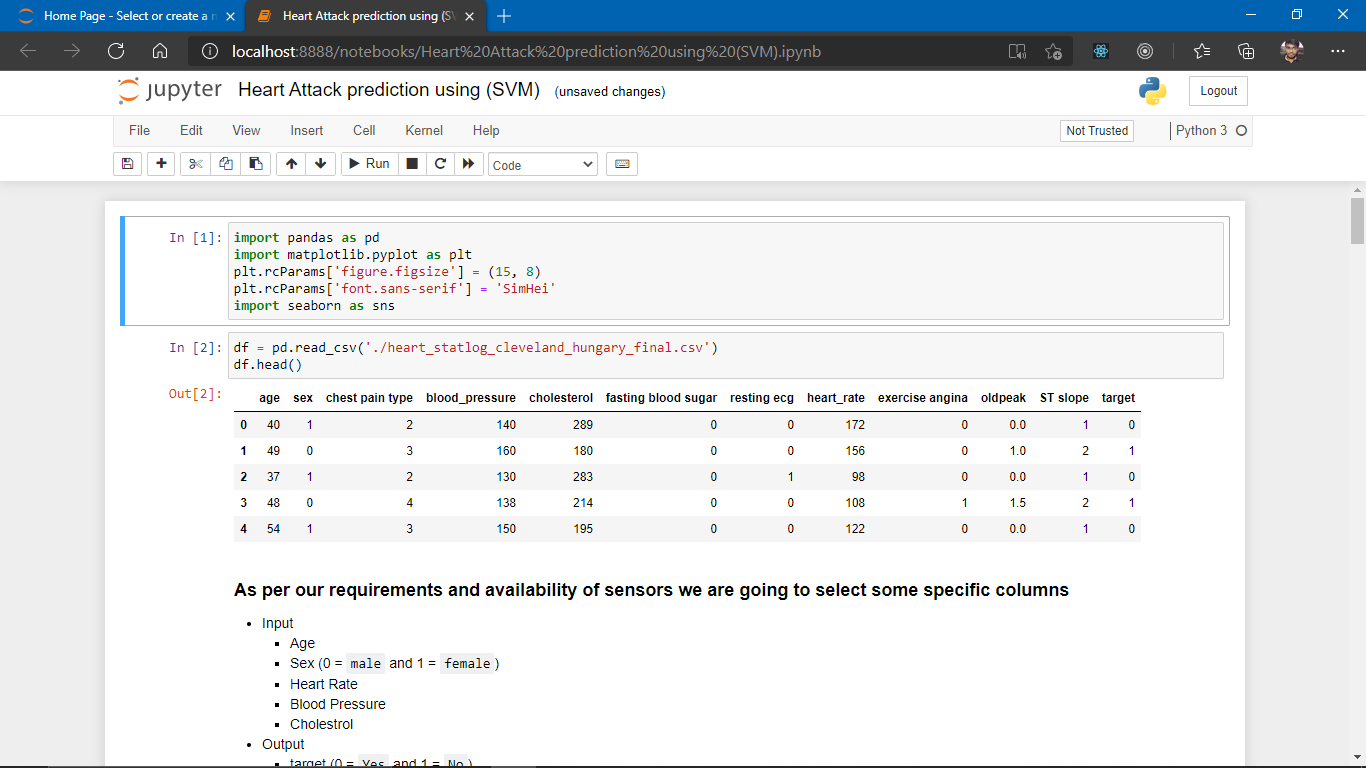
**Block diagram**

We are passing some user inputs from ardunio device as an API request which will later send to a Machine learning REST API. The output will be save to the Firebase database and then after the process user will get the data showed up on Arduino terminal or in a display.

These data will be saved on database and if he/she wants to visualize then they can through our web application where we will be using some nice charts to display the data.

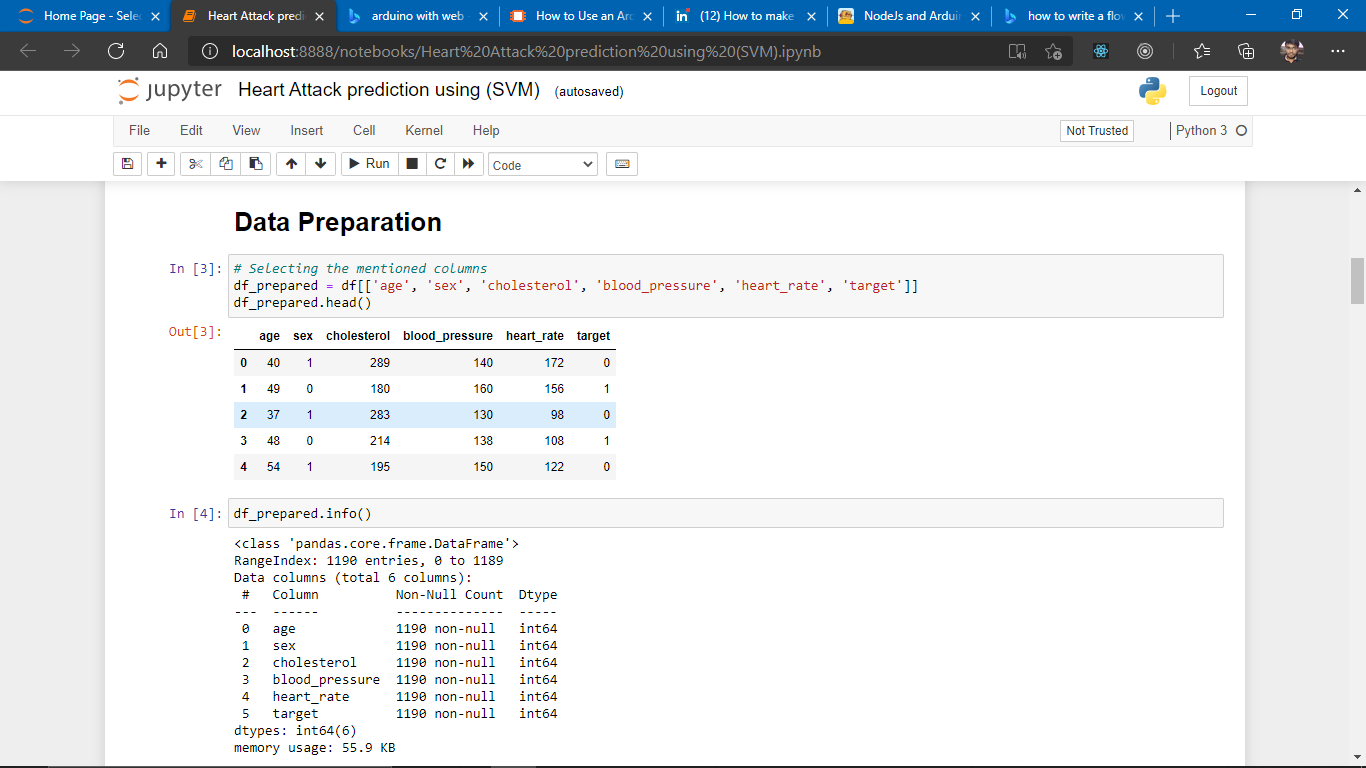
**Result Set:**

In the first step we gathered data from online resource to build our model to predict

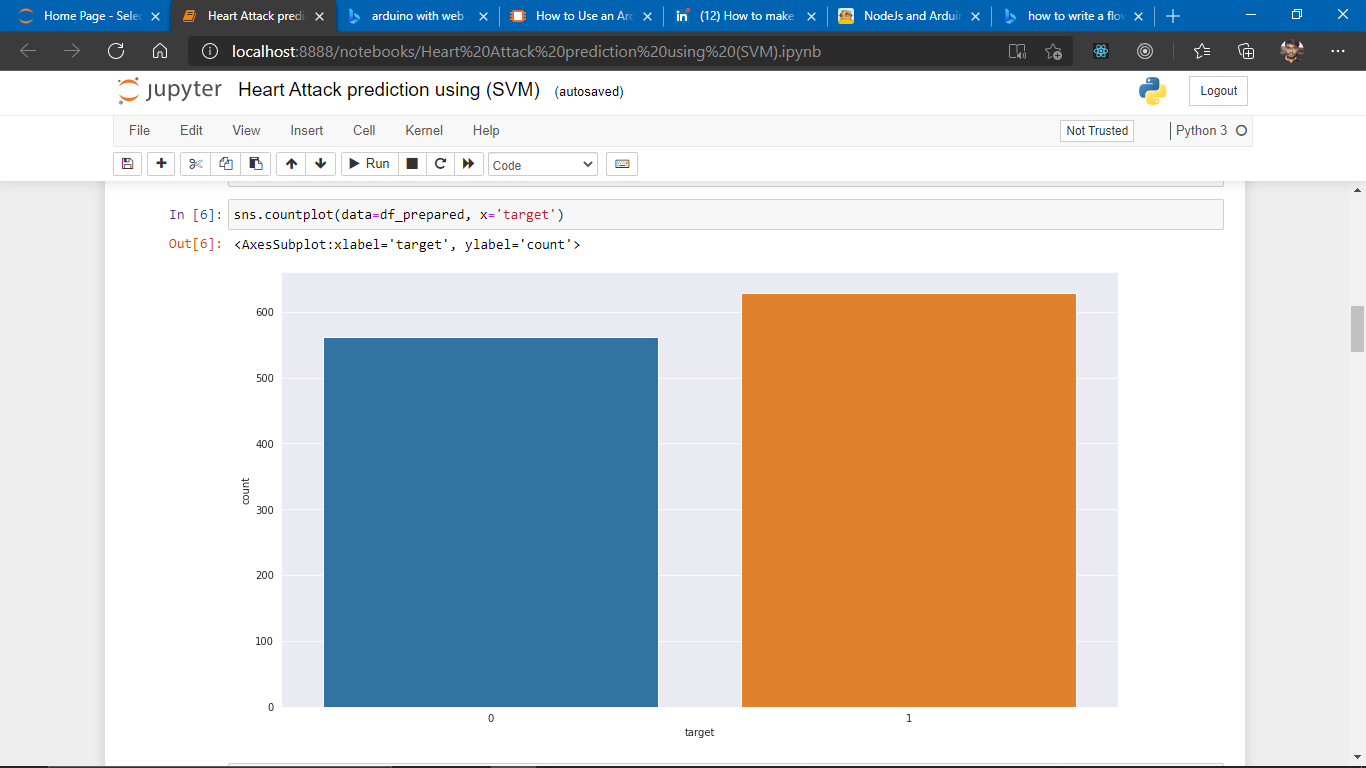
**Data From Kaggle.com**

In the beginning we first eliminated some attributes which we cannot calculate.

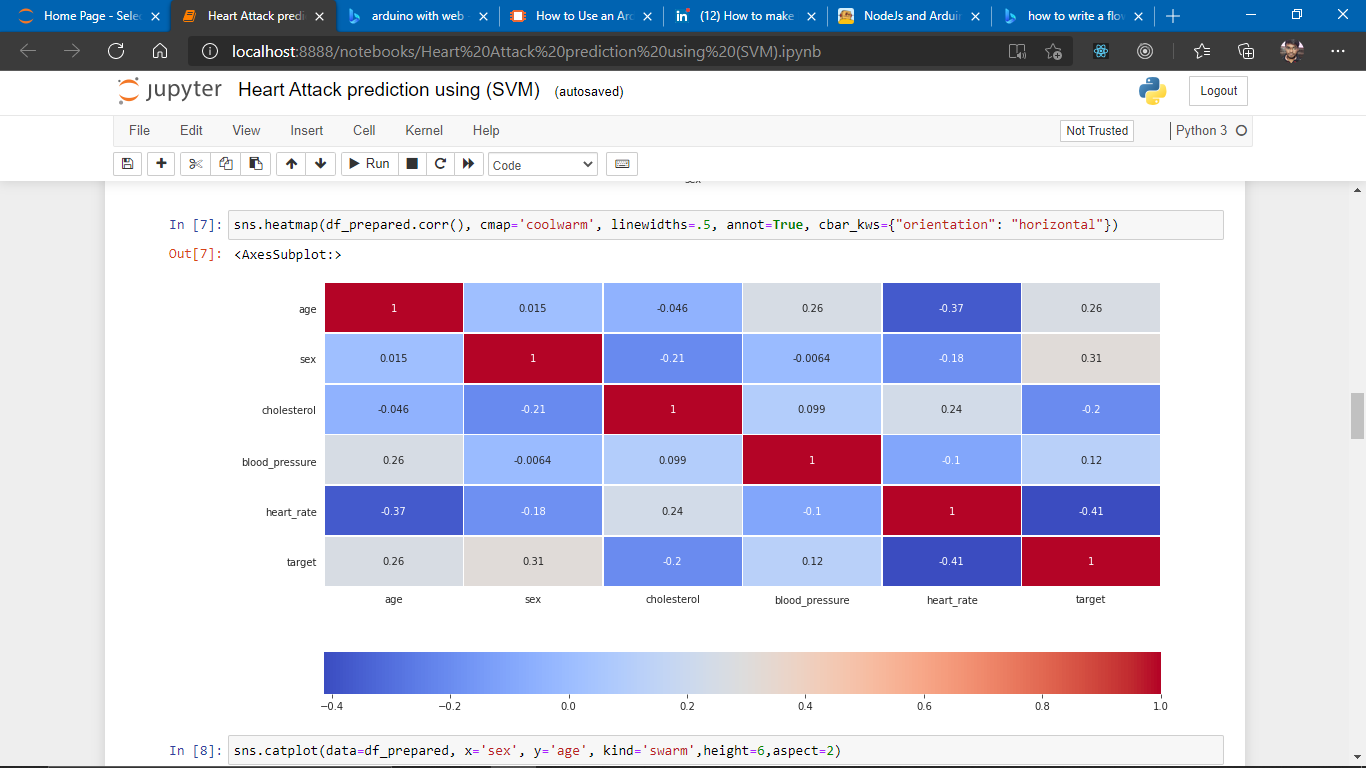
After reducing the datasets it became easy to analyze

**Data Preparation**

There is no missing data points and we have choose some data points which are relevant to us.

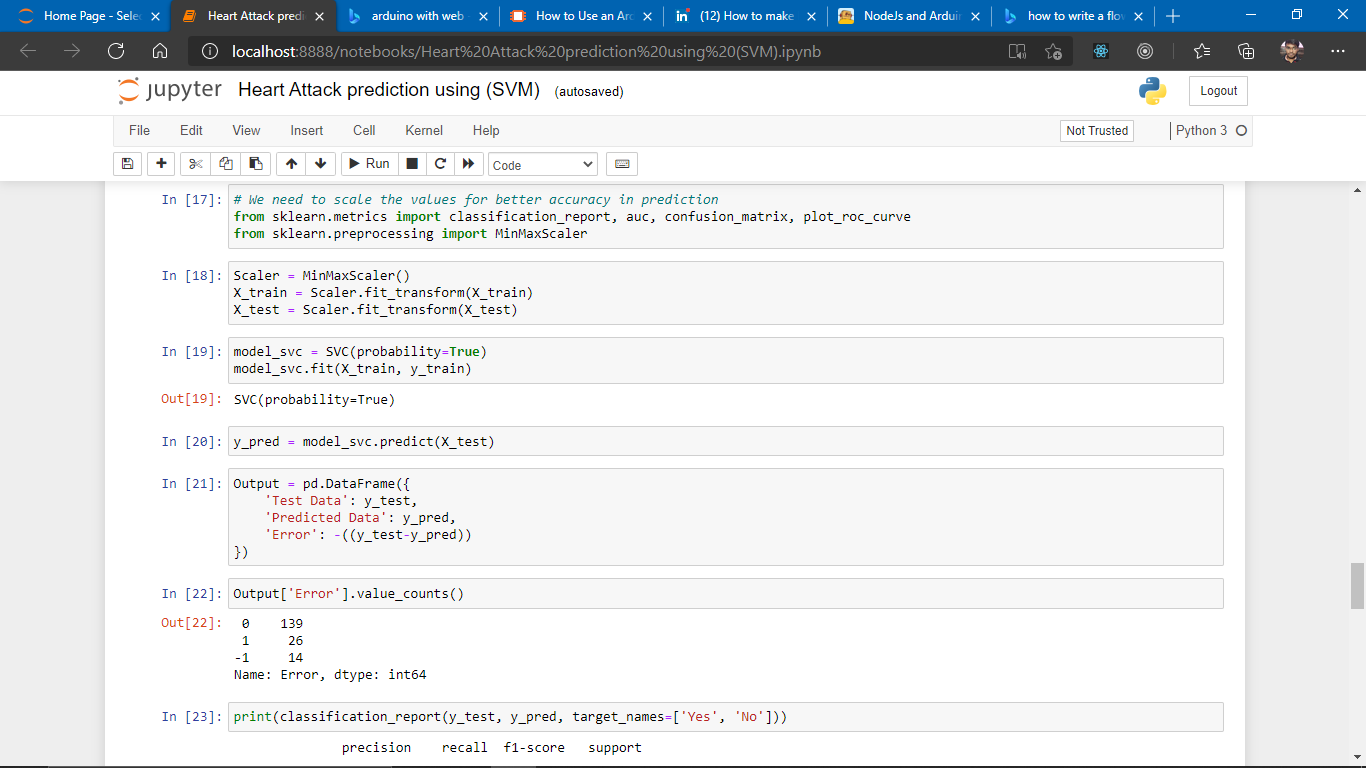
**Data Analysis**

By Performing analysis we found that data is not much gender biased it’s distribution is quite acceptable for prediction

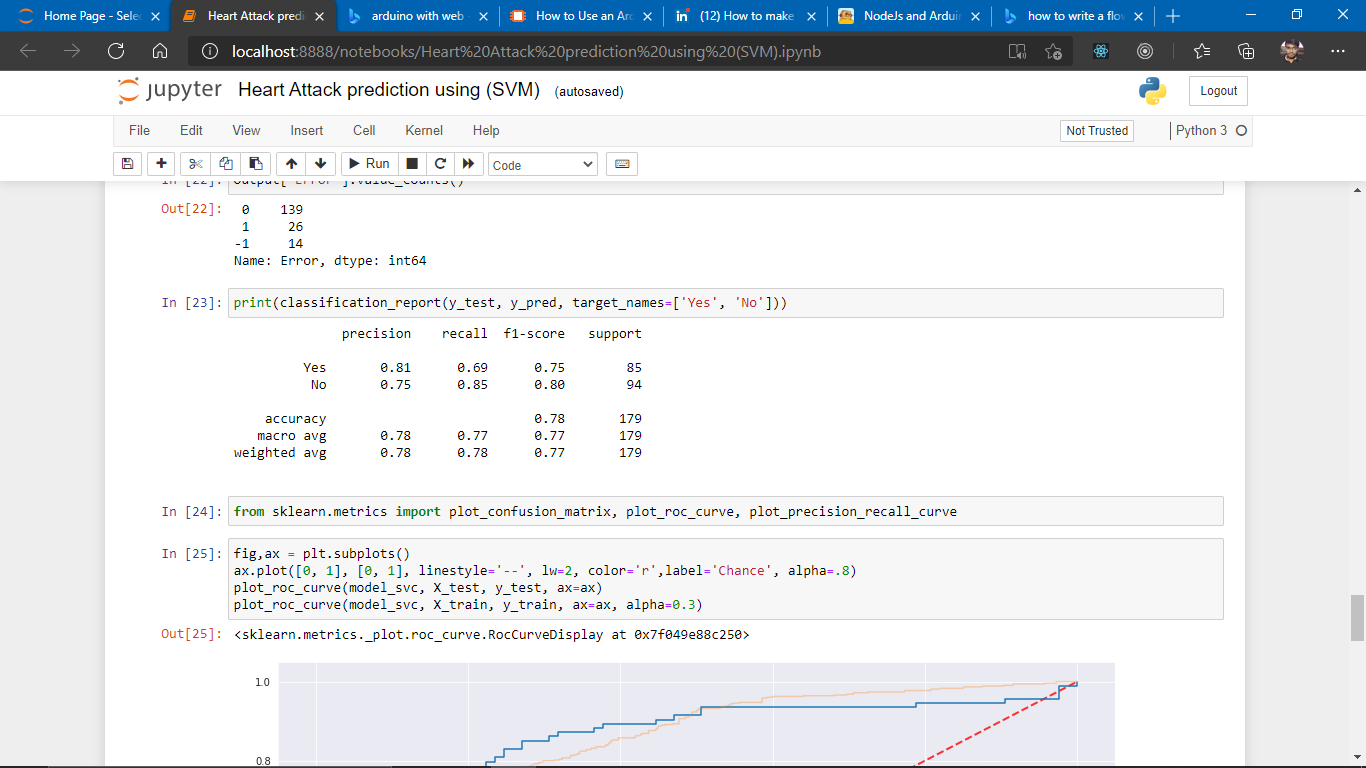
**Visualizing Correlations**

Heat-map calculated by taking Pearson's coefficient as [-1, 1]

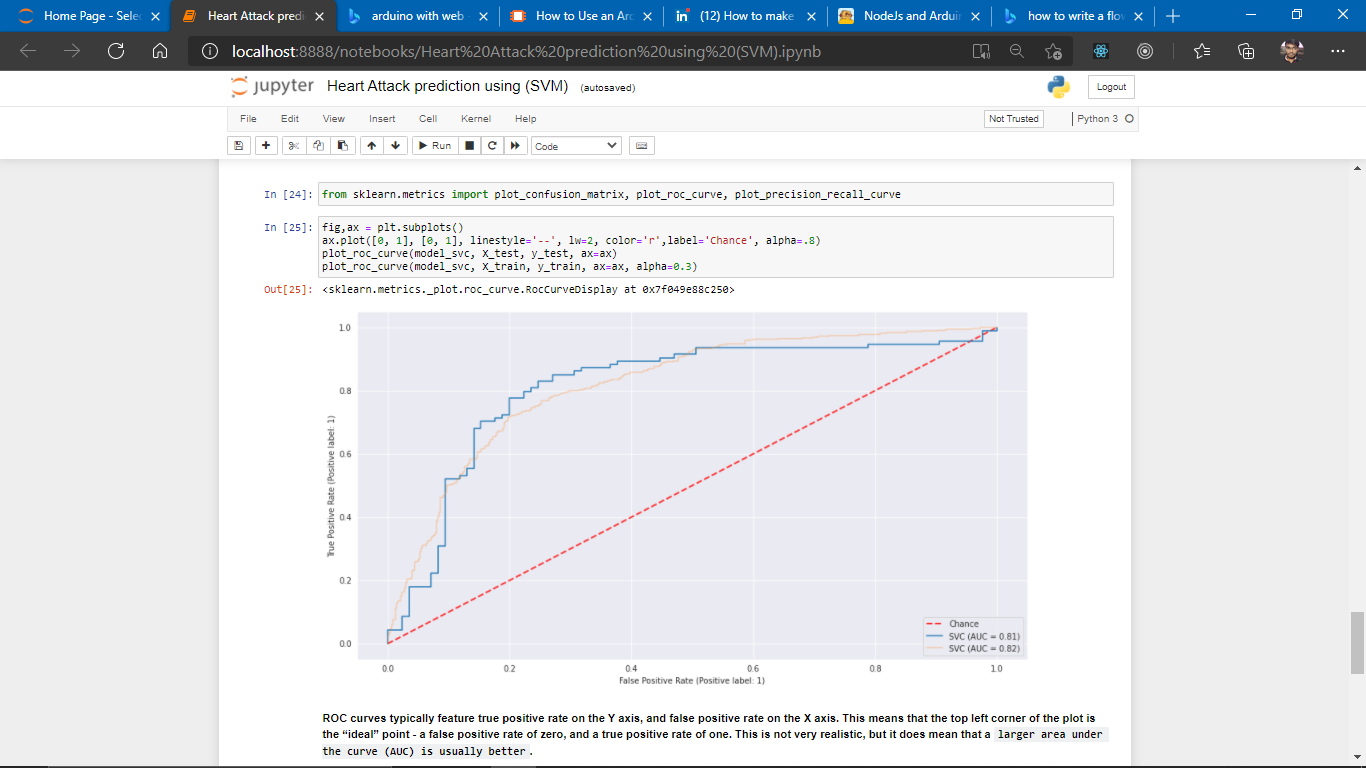
Using these types of graphs we can say which attribute is correlated more or less with other attribute or is there any particular relationship. **For example** blood pressure has a little amount of impact in heart attack.

**Choosing ML model Support Vector Machines (SVM)**

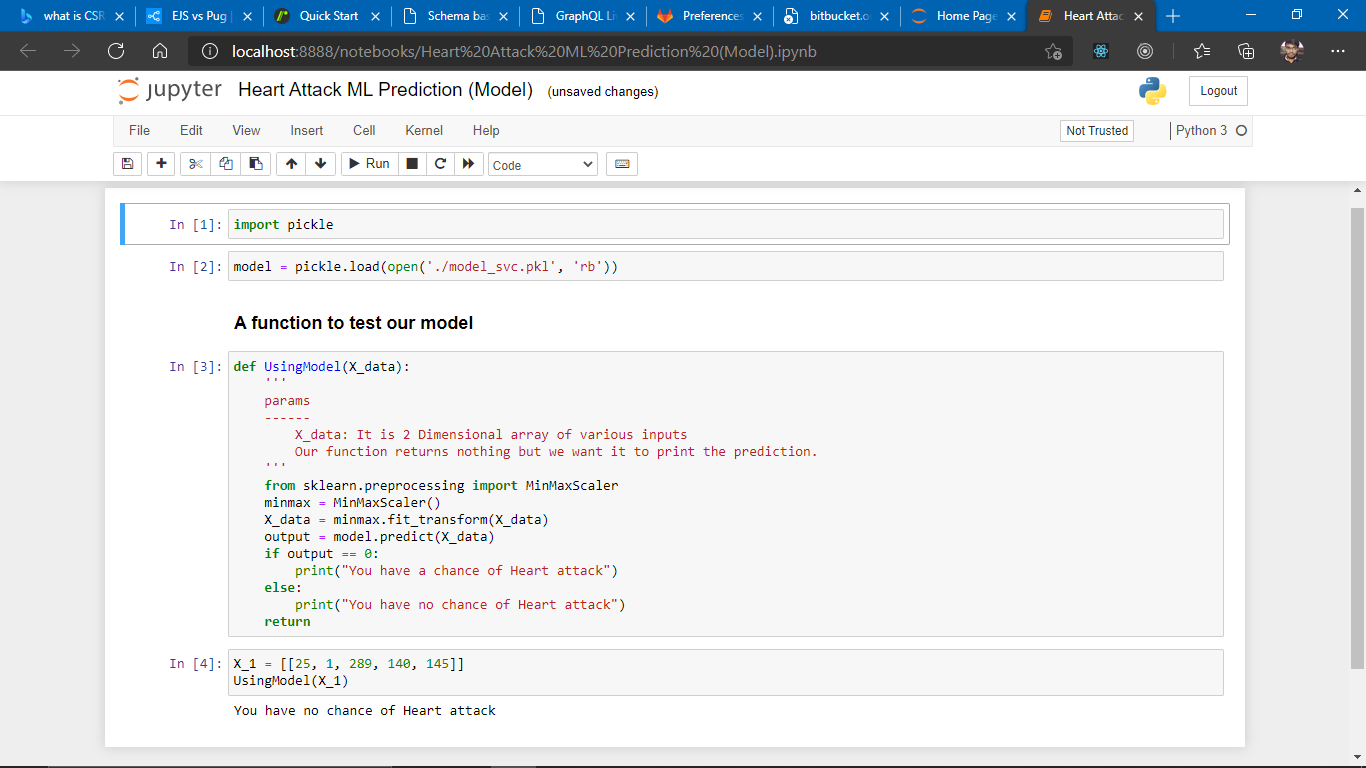
In order to work with data points we need to scales those points within the range of -1 and 1. For the ease in calculation and correct prediction

**Classification Report**

We can see that precision of predicting the attribute Y**es** is more than that of N**o.** And the accuracy of our model is also acceptable as we have less data right now.

**ROC-AUC Curve**

Our training and testing data are somewhat working the same as we are expecting from our calculation.

**Testing out the model we created**

Our model prediction accuracy is good enough as the data grows up model needs to be re-tuned.

**Discussion:**

In this project our goal is to predict the heart attack. Using Web Technologies, Machine learning and IoT concepts we have tried to achieve this goal.

We have three things in mind:-

1. Data is being entered by user in the device.

2. Data will be send to the web server using REST API web services and the output will be saved

to the database.

3. If a user wants he/she will be able to visualize there data because of dashboard functionality.

In the beginning we have small amount of data so our work is not that complicated but as soon the data will grow we would require to upgrade a lot of things as everything will become more complex to handle like the performance of model will decrease after every use. So In order to solve this problem I need to work on automated pipelines for data processing to training and testing of model. In the mean time we also need to work on scaling some aspect of server to optimize the load of many users.

There are more attributes we can take to build a sophisticated model for prediction but because of limitation in resources both financially and educationally. We cannot select them for example there is an attribute called cholesterol which cannot be measure by some simple sensor for that we need some more advanced tools. Which are out of reach for now.

Well in terms of hardware we can also choose from lots of resources like Raspberry Pi which is good for our project but the fact is that because of less accessibility we are using Arduino with a WiFi board. Specially the sensor we are using might not be well calibrated which is very important if the provided data is garbage then our prediction is useless and hence this project will become worthless.

In the whole process of sending and receiving data there is huge amount of network calls which our server and client needs to perform to send and receive data which will cause some latency and more data will be consume. We can avoid this problem by using web sockets.

If this project scales up then I will consider all the points and try to find an optimal solution to both hardware and software resources.

**Remaining work:**

* Hardware Development
* Arduino program
* Web Application (Frontend and Backend)
* Deployment

**References and Bibliography:**

Sample References:

1. M. Sanjeev Arulampalam, Simon Maskell, Neil Gordon, and Tim Clapp, “A tutorial on

particle filters for online nonlinear/non-Gaussian Bayesian tracking”, IEEE Transactions

on Signal Processing, Vol. 50, No. 2, pp:174-188, Feb’ 2002.

2. R. N. Banavar, J. L. Speyer, “Properties of Risk-Sensitive Filters/Estimators”, IEEE

Proceedings of Control Theory Application, Vol.145, No. 1, January 1998.

3. R. G. Brown, and P. Y. C. Hwang, Introduction to Random Signals and Applied

Kalman Filtering with Matlab Exercises and Solutions, 3rd Edition, John Wiley & Sons,

Inc, 1997.

4. Universal Description, Discovery and Integration, UDDI ; http://www.uddi.org; October

5, 2007.

**Appendix**

Machine learning Code

# We need to scale the values for better accuracy in prediction

from sklearn.metrics import classification\_report, auc, confusion\_matrix, plot\_roc\_curve

from sklearn.preprocessing import MinMaxScaler

Scaler = MinMaxScaler()

X\_train = Scaler.fit\_transform(X\_train)

X\_test = Scaler.fit\_transform(X\_test)

model\_svc = SVC(probability=True)

model\_svc.fit(X\_train, y\_train)

y\_pred = model\_svc.predict(X\_test)

print(classification\_report(y\_test, y\_pred, target\_names=['Yes', 'No']))

from sklearn.metrics import plot\_confusion\_matrix, plot\_roc\_curve, plot\_precision\_recall\_curve

fig,ax = plt.subplots()

ax.plot([0, 1], [0, 1], linestyle='--', lw=2, color='r',label='Chance', alpha=.8)

plot\_roc\_curve(model\_svc, X\_test, y\_test, ax=ax)

plot\_roc\_curve(model\_svc, X\_train, y\_train, ax=ax, alpha=0.3)

plot\_confusion\_matrix(model\_svc, X\_test, y\_test)

import pickle

# Saving the model

filename='model\_svc.pkl'

pickle.dump(model\_svc, open(filename, 'wb'))

df\_prepared.to\_csv('prepared\_data.csv', index=False)