**CHAPTER 1**

**INTRODUCTION**

* 1. **Motivation**

At present, when one suffers from a disease, then the person has to visit a doctor which is time consuming and costly too. Also, if the user is unable to contact a doctor or hospital it may be difficult for the user as the heart disease cannot be identified. So, if the process can be completed using an automated program which can save time as well as money, it could be easier for the patient making the process efficient. Hence, we propose a Heart Disease Prediction System using data mining techniques that analyzes the risk level of the patient. Smartphone utilization along with rising adoption of health applications, IoT, telehealth and other similar technologies are contributing to the market growth.

Heart Disease Predictor is a web based application that predicts the risk level of the disease of the user. This system has datasets collected from the UCI machine learning repository. With the help of Heart Disease Predictor, the user will be able to know the probability of the disease with the given values provided as input by the user.

As the use of internet is growing every day, people are always curious to know different new things. People always try to refer to the internet if any problem arises. People have access to internet than hospitals and doctors. People do not have immediate option when they suffer with a particular disease. So, this system can be helpful to the people as they have access to internet 24 hours.

The mission of this application is actually guiding people and assisting to reveal if they are seriously ill. After that, an individual could be transferred to a healthcare professional. This order of things essentially saves doctor’s time.

**1.2 Objective**

**1.2.1 General Objective**

* To implement Logistic Regression on our acquired dataset that predicts the disease risk probability as per the input of the user.

**1.2.2 Specific Objective**

* To develop web interface platform for the prediction of heart disease which consists of a form taking input from the user.
* The input is used to calculate the probability using the regression coefficients that were obtained from the analyst model.

**1.3 Algorithm used**

**1.3.1 Overview**

The logistic model (or logit model) is a widely used [statistical model](https://en.wikipedia.org/wiki/Statistical_model) that, in its basic form, uses a [logistic function](https://en.wikipedia.org/wiki/Logistic_function) to model a [binary](https://en.wikipedia.org/wiki/Binary_variable) [dependent variable](https://en.wikipedia.org/wiki/Dependent_variable); many more complex [extensions](https://en.wikipedia.org/wiki/Logistic_regression#Extensions) exist. In [regression analysis](https://en.wikipedia.org/wiki/Regression_analysis), logistic regression (or logit regression) is [estimating](https://en.wikipedia.org/wiki/Estimation_theory) the parameters of a logistic model.

Mathematically, a binary logistic model has a dependent variable with two possible values, such as pass/fail, win/lose, alive/dead or healthy/sick; these are represented by an [indicator variable](https://en.wikipedia.org/wiki/Indicator_variable), where the two values are labeled "0" and "1". In the logistic model, the [log-odds](https://en.wikipedia.org/wiki/Log-odds) (the [logarithm](https://en.wikipedia.org/wiki/Logarithm) of the [odds](https://en.wikipedia.org/wiki/Odds)) for the value labeled "1" is a [linear combination](https://en.wikipedia.org/wiki/Linear_function_(calculus)) of one or more [independent variables](https://en.wikipedia.org/wiki/Independent_variable) ("predictors"); the independent variables can each be a binary variable (two classes, coded by an indicator variable) or a [continuous variable](https://en.wikipedia.org/wiki/Continuous_variable) (any real value). The corresponding [probability](https://en.wikipedia.org/wiki/Probability) of the value labeled "1" can vary between 0 (certainly the value "0") and 1 (certainly the value "1"), hence the labeling; the function that converts log-odds to probability is the logistic function.

Logistic regression is used in various fields, including machine learning, most medical fields, and social sciences. For example, the Trauma and Injury Severity Score ([TRISS](https://en.wikipedia.org/wiki/TRISS)), which is widely used to predict mortality in injured patients, was originally developed by Boyd et al. using logistic regression.

A regression coefficient describes the size and direction of the relationship between a predictor and the response variable. Coefficients are the numbers by which the values of the term are multiplied in a regression equation.

**1.4 Scope and Limitations**

**1.4.1 Scope**

This project aims to provide a web platform to predict the occurrence of a heart disease on the basis of given inputs. The user can give values for various symptoms and can find the predicted value with their probabilistic figures.

**1.4.2 Limitations**

* The limitations of this project are:
* Disease Predictor does not recommend medications for the heart disease. Past history of the heart disease has not been considered

**CHAPTER 2**

**LITERATURE SURVEY**

**2.1 Overview**

Heart disease is a term that assigns to a large number of medical conditions related to heart. These medical conditions describe the abnormal health conditions that directly influence the heart and all its parts. Heart disease is a major health problem in today’s time. A heart disease is caused due to narrowing or blocking of coronary arteries. This is caused by the deposition of fat on the inner walls of the arteries and also due to build up cholesterol. Nowadays, in the world Heart disease is the major cause of deaths. The World Health Organization (WHO) has estimated that 12 million deaths occur worldwide, every year due to the Heart diseases. There are number of factors which increases risk of Heart disease.

The risk factor for heart disease family history are: -

Most people know that the heat disease can run in families. That if anybody has a family history of heart disease, he/she may be at greater risk for heart attack, stroke and other heard diseases.

**Smoking: -** smoking is major cause of heart attack, stroke and other peripheral arterial disease. Nearly 40% of all people who die from smoking tobacco do so due of heart and blood vessel diseases. A smoker’s risk of heart attack reduces rapidly after only one year of not smoking.

**Cholesterol: -** abnormal levels of lipids (fats) in the blood are risk factor of heart diseases. Cholesterol is a soft, waxy substance found among the lipids in the bloodstream and in all the body’s cells. High level of triglyceride (most common type of fat in body) combined with high levels of LDL (low density lipoprotein) cholesterol speed up atherosclerosis increasing the risk of heart diseases.

**High blood pressure:** - High blood pressure also known as HBP or hypertension is a widely misunderstood medical condition. High blood pressure increase the risk of the walls of our blood vessels walls becoming overstretched and injured. Also increase the risk of having heart attack or stroke and of developing heart failure, kidney failure and peripheral vascular disease.

**Obesity:-**the term obesity is used to describe the health condition of anyone significantly above his or her ideal healthy weight. Being obese puts anybody at a higher risk for health problem such as heart disease, stroke, high blood pressure, diabetes and more.

**Lack of physical exercise:** -lack of exercise is a risk factor for developing coronary artery disease (CAD). Lack of physical exercise increases the risk of CAD, because it also increases the risk for diabetes and high blood pressure.

The symptoms for heart disease are:

The symptoms include tightness or pain in the chest, back or arms, neck, as well as fatigue, light headedness, abnormal heartbeat and anxiety. Women are more likely to have a typical symptoms than men.

a) Pain area: area between shoulders blades, arm, chest, jaw, left arm or upper abdomen.

b) Pain types: can be crushed, like a clenched first in the chest, radiating from the chest, sudden in the chest, or mild.

c) Pain circumstances: may occur during rest

d) Whole body: dizziness, fatigue, light-headedness, clammy skin, cold sweat, or sweating. e) Gastrointestinal: heartburn, indigestion, nausea or vomiting.

f) Chest: discomfort, fullness or tightness.

g) Neck: discomfort or tightness.

h) Arm: discomfort or tightness.

**2.2 Machine Learning algorithms**

Machine

Learning

Techniques

Supervised

Regression

Classification

Binary

Multiclass

Unsupervised

Clustering

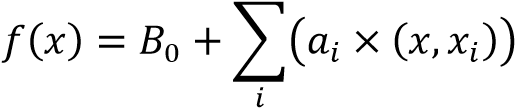
Dimensionality

Reduction

1. **Supervised learning:** In supervised learning, answer is already known. The main idea is to design a model that can predict the answers for unknown instances. We give the machine the data, with inputs and outputs (the answers), and let it learn from the relationships between them. We can further divide supervised learning techniques into regression and classification based on the variable being predicted. If the predicted value is continuous, it is Regression problem. Otherwise, if the variable to be predicted is one of various independent categories, called classes, it is known as Classification problem. If there are two classes, then it is binary classification and if we have multiple classes then there is multiclass classification problem. Some of the supervised algorithms are below:

* **Support Vector Machine**

It constructs a hyper plane as shown in Figure 2 i.e. a plane in an infinite dimension plane to classify the training data points into clearly demarcated classes. The construction of an optimal decision plane for classification requires minimizing the error function. The shape of the error function becomes the foundation for further classification of these algorithms in the broad categories of linear, polynomial, sigmoid and radial SVMs. So, putting in simple terms the philosophy of SVM is to obtain an optimal hyper plane for data points which are linearly separable. Support vectors actually refer to the data points that are closest to the demarcating surface which are hence tricky to classify. The metric that alludes to the optimality of a hyper plane is the margin around the hyper plane. So, the problem transitions into that of an optimization one. As established the maximum margin classifier learnt and derived from the training data would lead us to optimal hyper plane. This is achieved by transforming the maximal margin classifier as the inner product (sum of multiplication of pair values) of two given data points rather than the data points. The general kernel function could then be defined as follows:



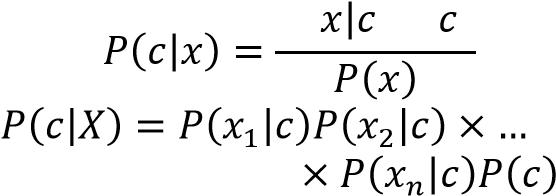
Here 𝑥 is the new input vector and the coefficients 𝐵0 and 𝑎𝑖 must be estimated from training data.

* **Decision Tree**

Decision tree are supervised method used for the prediction of categorical as well as numerical value..They represents the data instances along with their class label in the form of a tree. A set of rules can be deduced from the tree which can be used to classify the unknown data record to its output value. A test on an attribute is performed on the internal node. The result of the test is depicted by the branch of tree and class label are present at the leaf node. In this technique, the whole data set or the whole collection of sample points in split into two or more homogenous classes. The split is established from the parameter or the factor which is determined to be the best splitter or differentiator.

* **Naïve Bayes:**

Rather than a single classifier it is a combination of multiple classifiers all working on the basic Naïve Bayes principle of independent features. Hence each feature is assumed to be independent and autonomous contributing individually to the training data point’s probability of belonging to a particular class.As per the Bayes theorem,

 𝑃( )𝑃( )

Here,

𝑃(𝑐|𝑥)is the posterior probability of class given predictor

𝑃(𝑐)is the prior probability of class 𝑃(𝑥|𝑐)is the likelihood probability of predictor given class 𝑃(𝑋)is the prior probability of predictor

* **Artificial Neural Network**

These are used to model/simulate the distribution, functions or mappings among variables as modules of a dynamic system associated with a learning rule or a learning algorithm. The modules here simulate neurons in nervous system and hence ANN collectively refers to the neuron simulators and their synapsis simulating interconnections between these modules in different layers [6]. The defining aspect of an ANN is the function implemented at each neuron and the learning algorithm for the dynamic weights assigned to the interconnections among neurons. What makes ANN stand apart is its ability to simulate human thought process coupled with continuous learning, growth and evolution. Also, it is capable of handling large number of parameters and large set of data with noise and yet achieves high accuracy.

1. **Unsupervised learning:** In unsupervised learning, our aim is to find unknown trends. The data has no associated labels, but we want to organize the data into groups or clusters. Unsupervised learning techniques are further classified as Cluster Analysis and Dimensionality Reduction. In Cluster Analysis, data is grouped according to similarities or distances between them. In Dimension Reduction, duplicated or unnecessary variables are removed to produce a smaller subset of the original data.

**2.3 Dataset**

**2.3.1 Description**

This directory contains 4 databases concerning heart disease diagnosis. All attributes are numeric-valued. The data was collected from the four following locations:

1. Cleveland Clinic Foundation (cleveland.data)

2. Hungarian Institute of Cardiology, Budapest (hungarian.data)

3. V.A. Medical Center, Long Beach, CA (long-beach-va.data)

4. University Hospital, Zurich, Switzerland (switzerland.data)

This database contains 76 attributes, but all published experiments refer to using a subset of 14 of them. In particular, the Cleveland database is the only one that has been used by ML researchers to this date. The "goal" field refers to the presence of heart disease in the patient. It is integer valued from 0 (no presence) to 4. Experiments with the Cleveland database have concentrated on simply attempting to distinguish presence (values 1,2,3,4) from absence (value 0).

Number of Instances:

Database: # of instances:

Cleveland: 303

Hungarian: 294

Switzerland: 123

Long Beach VA: 200

Number of Attributes: 76 (including the predicted attribute)

Missing Attribute Values: Several. Distinguished with value -9.0.

Class Distribution:

Database: 0 1 2 3 4 Total

Cleveland: 164 55 36 35 13 303

Hungarian: 188 37 26 28 15 294

Switzerland: 8 48 32 30 5 123

Long Beach VA: 51 56 41 42 10 200

**2.3.2 Attributes information**

14 used are-

1. Age (in years)
2. sex - (1 = male; 0 = female)
3. Cp - chest pain type

Value 1: typical angina

Value 2: atypical angina

Value 3: non-anginal pain

Value 4: asymptomatic

1. trestbps - resting blood pressure (in mm Hg on admission to the hospital)
2. chol serum cholestoral in mg/dl
3. fbs - (fasting blood sugar > 120 mg/dl) (1 = true; 0 = false)
4. restecg - resting electrocardiographic results

Value 0: normal

Value 1: having ST-T wave abnormality (T wave inversions and/or ST elevation or depression of > 0.05 mV)

Value 2: showing probable or definite left ventricular hypertrophy by Estes' criteria

1. thalach - maximum heart rate achieved
2. exang - exercise induced angina (1 = yes; 0 = no)
3. oldpeak - ST depression induced by exercise relative to rest
4. slope: the slope of the peak exercise ST segment

Value 1: upsloping

Value 2: flat

Value 3: downsloping

1. ca - number of major vessels (0-3) colored by flourosopy
2. thal

3 = normal

6 = fixed defect

7 = reversable defect

1. num - the predicted attribute

**2.4 Past usage of the dataset**

1. Detrano, R., Janosi, A., Steinbrunn, W., Pfisterer, M., Schmid, J., Sandhu,~S., Guppy,~K., Lee, S., & Froelicher, V. (1989).

International application of a new probability algorithm for the diagnosis of coronary artery disease. American Journal of Cardiology, 304--310.

International Probability Analysis

Results in percent accuracy: (for 0.5 probability threshold)

Data Name: CDF CADENZA

- Hungarian 77 74

- Long beach 79 77

- Swiss 81 81

- Approximately a 77% correct classification accuracy with a logistic-regression-derived discriminant function

2. David W. Aha & Dennis Kibler

- Instance-based prediction of heart-disease presence with the Cleveland database

- NTgrowth: 77.0% accuracy

- C4: 74.8% accuracy

3. John Gennari

- Gennari, J.H., Langley, P, \& Fisher, D. (1989). Models of incremental concept formation.

- Results: The CLASSIT conceptual clustering system achieved a 78.9% accuracy on the Cleveland database.

**Chapter 3**

**SYSTEM ANALYSIS**

**3.1 Problems with existing system**

The data in hospitals are confidential. The data allowed to collect mostly consists of name, age and sex which is not sufficient for prediction. Heart diseases depend on a lot of attributes such as cholesterol, chest pain types, fasting blood pressure etc. Collecting of these data is against the hospitals rules. We therefore had to analyse the data from a dataset repository (here, UCI repository) which had datasets from four different hospitals from four different places. This increased the diversity in the data. Although the data was finally collected, it had a lot of missing values, which was then cleaned for further analysis.

**3.2 Proposed system**

For the data, we chose to download from UCI machine learning repository, as we could not collect from hospitals. To avoid overfitting, we divided the data into training and testing sets and fed it into the Logistic Regression model for analysis purpose. The regression coefficients then obtained from the model are calculated with the input provided by the user using the prediction formula.

**3.3 Processed logic**

The algorithm we used in this project is Logistic Regression. Logistic regression is the appropriate regression analysis to conduct when the dependent variable is dichotomous (binary).  Like all regression analyses, the logistic regression is a predictive analysis.  Logistic regression is used to describe data and to explain the relationship between one dependent binary variable and one or more nominal, ordinal, interval or ratio-level independent variables.

The dependent variable should be dichotomous in nature (e.g., presence vs. absent).

There should be no outliers in the data, which can be assessed by converting the continuous predictors to standardized scores, and removing unnecessary values. There should be no high correlations (multicollinearity) among the predictors.  This can be assessed by a correlation matrix among the predictors. Tabachnick and Fidell (2013) suggest that as long correlation coefficients among independent variables are less than 0.90 the assumption is met.

**3.3.1 Formulae used**

Prediction formula-

Y=w[0] + w[1] \* x[1] + . . . + w[p] \* x[p] + b > 0

where w – slope (or coefficient),

b – offset ( or intercept),

w and b are learned parameters

Y – prediction (or decision)

To obtain predicted value between 0-1, we use the sigmoid function -

H=1/(1+e-y)

Where H – output value,

Y – predicted value

**3.3.2 Normalization techniques**

Many machine learning algorithms attempt to find trends in the data by comparing features of data points. However, there is an issue when the features are on drastically different scales. The goal of normalization is to make every datapoint have the same scale so each feature is equally important. The image below shows the same house data normalized using min-max normalization.

**Min-max normalization**

Min-max normalization is one of the most common ways to normalize data. For every feature, the minimum value of that feature gets transformed into a 0, the maximum value gets transformed into a 1, and every other value gets transformed into a decimal between 0 and 1.

For example, if the minimum value of a feature was 20, and the maximum value was 40, then 30 would be transformed to about 0.5 since it is halfway between 20 and 40. The formula is as follows:

Minmax = (value – min)/(max – min)

**Z – score normalization**

Z-score normalization is a strategy of normalizing data that avoids this outlier issue. The formula for Z-score normalization is below:

z = (x – μ) / σ

where x – sample mean

μ – population mean

σ – standard deviation

Here, μ is the mean value of the feature and σ is the standard deviation of the feature. If a value is exactly equal to the mean of all the values of the feature, it will be normalized to 0. If it is below the mean, it will be a negative number, and if it is above the mean it will be a positive number. The size of those negative and positive numbers is determined by the standard deviation of the original feature. If the unnormalized data had a large standard deviation, the normalized values will be closer to 0.

**Chapter 4**

**DESIGN ANALYSIS**

**4.1 Architecture Design**

This secure data analytics approach is divided into five modules involving Collecting Data, Analyzing Data, Data Wrangling, Train and Test, Accuracy Check and Result. It uses Logistic Regression to predict the heart disease risk percentage. The necessary datasets are taken from UCI repository for training purpose.

****

*Fig. 4.1. Block Diagram of Proposed System*

**Collecting Data:**

It consists of importing the necessary libraries (such as pandas, numpy, matplotlib) and importing the dataset that was downloaded from UCI repository.

**Analyzing Data:**

Here, histograms are plotted against various attributes of the dataset for graphical analyzing purposes and data cleaning. Any unnecessary attribute that does not contribute to the model is removed from the dataset.

**Data Wrangling:**

Cleaning of data is to be done using dummies, where any categorical values are converted to discrete values.

Data wrangling basically consists of converting categorical values into discrete values. The main distinction is between [continuous variables](https://en.wikipedia.org/wiki/Continuous_variable)(such as income, age and [blood pressure](https://en.wikipedia.org/wiki/Blood_pressure)) and [discrete variables](https://en.wikipedia.org/wiki/Discrete_variable) (such as sex or race). Discrete variables refer to more than two possible choices and are coded using [dummy variables](https://en.wikipedia.org/wiki/Dummy_variable_(statistics)) i. e separate explanatory variables taking the value 0 or 1 are created for each possible value of the discrete variable, with a 1 meaning "variable does have the given value" and a 0 meaning "variable does not have that value".

**Train and Test Data:**

The model is divided into training set and testing set on which Logistic Regression is carried out.

**Accuracy Check:**

We can calculate the accuracy to check how accurate our model is.

**4.2 Diagrams**

**4.2.1 Class Diagram**

The class diagram is a static diagram. It represents the static view of an application. Class diagram is not only used for visualizing, describing and documenting different aspects of a system but also for constructing executable code of the software application



*Fig.4.2: Class Diagram of the app*

**4.2.2 Use Case Diagram**

Use Case diagrams identify the functionality provided by the system (use cases), the users who interact with the system (actors), and the association between the users and the functionality. Use Cases are used in the Analysis phase of software development to articulate the high-level requirements of the system. The primary goals of Use Case diagrams include:

• Providing a high-level view of what the system does

• Identifying the users ("actors") of the system

• Determining areas needing human-computer interfaces



*Fig.4.3: Use Case Diagram of the app*

**4.2.3 Sequence Diagram**

Sequence diagram emphasizes on time sequence of messages and collaboration diagram emphasizes on the structural organization of the objects that send and receive message



*Fig.4.4: Sequence Diagram from user of the app*



*Fig.4.5: Sequence Diagram for analyst of the app*

**4.2.4 Activity Diagram**

The Activity Diagram forms effective while modelling the functionality of the system. Hence this diagram reflects the activities, the types of flows between these activities and finally the response of objects to these activities.



*Fig.4.6: Activity Diagram of the app*

**4.2.5 Component Diagram**

Component diagrams are different in terms of nature and behaviour. Component diagrams are used to model physical aspects of a system. Component diagrams are used to visualize the organization and relationships among components in a system. These diagrams are also used to make executable systems.



*Fig.4.8: Component Diagram of the app*

**4.2.6 Deployment Diagram**

Deployment diagram represents the deployment view of a system. It is related to the component diagram. Because the components are deployed using the deployment diagrams. A deployment diagram consists of nodes. Nodes are nothing but physical hardware used to deploy the application.

**

*Fig 4.9 Deployment Diagram of the app*

**Chapter 5**

**IMPLEMENTATION**

## 5.1 Introduction

Implementation is the stage of the project when the theoretical design is turned out into a working system. Thus, it can be considered to be the most critical stage in achieving a successful new system and in giving the user, confidence that the new system will work and be effective. The implementation stage involves careful planning, investigation of the existing system and its constraints on implementation, designing of methods to achieve changeover and evaluation of changeover methods.

**5.2 Modules**

There are two main modules, they are:

**5.2.1 Module – 1: Analyst model**

The analyst model performs logistic regression on the data that is provided to it. Here, data wrangling is carried out and is trained by logistic regression. The regression coefficients are then calculated which describes the size and direction of the relationship between a predictor and the response variable.

**5.2.2 Module – 2: Webpage**

The webpage basically consists of a form that takes input from the user and calculates the risk percentage using the logistic regression prediction formula with the help of the regression coefficients. Coefficients are the numbers by which the values of the term are multiplied in a regression equation.

## 5.3 Programming Code

**Convert\_ssv\_to\_csv.py**

**import** sys  
  
file1 = sys.argv[1]+**'.ssv'**file2 = sys.argv[1]+**'.csv'**input\_file = open(file1, **'r'**)  
output\_file = open(file2, **'w'**)  
input\_record = 0  
**for** line **in** input\_file:  
**if** line.strip():  
input\_record += 1  
 line = line.replace(**" "**, **","**)  
output\_file.write(line)  
input\_file.close()  
output\_file.close()  
print (**"\n %i records transferred from %s to %s"** % (input\_record, file1, file2))

**Join\_files.py**

input\_files = [**"cleveland.csv"**, **"hungarian.csv"**, **"long\_beach\_va.csv"**,  
**"switzerland.csv"**]  
output\_file = **"heart\_disease\_all.csv"**data\_dir = **"C:/Users/Mohammed/HeartDisease/"**output = open(data\_dir+output\_file, **'w'**)  
total\_input = 0  
total\_output = 0  
**for** input\_file**in** input\_files:  
 input = open(data\_dir+input\_file, **'r'**)  
nlines\_through = 0  
nlines\_total = 0  
**for** line **in** input:  
total\_input += 1  
nlines\_total += 1  
**if** (**"-9" not in** line) **and** (**"?" not in** line):  
features\_list = line.split(**","**)  
features\_list = [float(item) **for** item **in** features\_list[0:14]]  
corrected\_line = **","**.join(map(str, features\_list))  
output.write(corrected\_line+**"\n"**)  
total\_output += 1  
nlines\_through += 1  
 print (**"Transferred %i out of %i records from %s"** % (nlines\_through, nlines\_total, input\_file))  
input.close()  
print (**"Total records read in: %i"** % total\_input)  
print (**"Total records written out to %s: %i"** % (output\_file, total\_output))  
output.close()

**AnalystModel.ipynb**

**#impoting libraries**

%matplotlib inline

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

from collections import Counter

from pprint import pprint

**#Reading the data from CSV file using pandas and creating Pandas dataframe.**

columns = ["age", "sex", "cp", "restbp", "chol", "fbs", "restecg", "thalach", "exang", "oldpeak", "slope", "ca", "thal", "num"]

table = pd.read\_table ("C:/Users/Mohammed/HeartDisease/heart\_disease\_all.csv", sep=',', header=None, names=columns)

**#Viewing first few rows of dataset**

table.head()

**#Computing mean,count,etc. characteristics**

table.describe()

**#Checking feature information**

table.info()

**#Plotting histograms**

fig, axes = plt.subplots( nrows=5, ncols=3, figsize=(15,20) )

plt.subplots\_adjust( wspace=0.20, hspace=0.20, top=0.94 )

plt.suptitle("Heart Disease Histograms", fontsize=20)

axes[0,0].hist(table.age)

axes[0,0].set\_xlabel("Age (years)")

axes[0,0].set\_ylabel("Number of Patients")

axes[0,1].hist(table.sex)

axes[0,1].set\_xlabel("Sex (0=female,1=male)")

axes[0,1].set\_ylabel("Number of Patients")

axes[0,2].hist(table.cp,bins=4,range=(0.5,4.5),rwidth=0.80)

axes[0,2].set\_xlim(0.0,5.0)

axes[0,2].set\_xlabel("Type of Chest Pain [cp]")

axes[0,2].set\_ylabel("Number of Patients")

axes[1,0].hist(table.restbp)

axes[1,0].set\_xlabel("Resting Blood Pressure [restbp]")

axes[1,0].set\_ylabel("Number of Patients")

axes[1,1].hist(table.chol)

axes[1,1].set\_xlabel("Serum Cholesterol [chol]")

axes[1,1].set\_ylabel("Number of Patients")

axes[1,2].hist(table.fbs)

axes[1,2].set\_xlabel("Fasting Blood Sugar [fbs]")

axes[1,2].set\_ylabel("Number of Patients")

axes[2,0].hist(table.restecg)

axes[2,0].set\_xlabel("Resting Electrocardiography [restecg]")

axes[2,0].set\_ylabel("Number of Patients")

axes[2,1].hist(table.thalach)

axes[2,1].set\_xlabel("Maximum Heart Rate Achieved [thalach]")

axes[2,1].set\_ylabel("Number of Patients")

axes[2,2].hist(table.exang)

axes[2,2].set\_xlabel("Exercise Induced Angina [exang]")

axes[2,2].set\_ylabel("Number of Patients")

axes[3,0].hist(table.oldpeak)

axes[3,0].set\_xlabel("Exercise Induced ST Depression [oldpeak]")

axes[3,0].set\_ylabel("Number of Patients")

axes[3,1].hist(table.slope)

axes[3,1].set\_xlabel("Slope of Peak Exercise ST Segment [slope]")

axes[3,1].set\_ylabel("Number of Patients")

axes[3,2].hist(table.ca,bins=4,range=(-0.5,3.5),rwidth=0.8)

axes[3,2].set\_xlim(-0.7,3.7)

axes[3,2].set\_xlabel("Major Vessels colored by Fluoroscopy [ca]")

axes[3,2].set\_ylabel("Number of Patients")

axes[4,0].hist(table.thal)

axes[4,0].set\_xlabel("Thal")

axes[4,0].set\_ylabel("Number of Patients")

axes[4,1].hist(table.num,bins=5,range=(-0.5,4.5),rwidth=0.8)

axes[4,1].set\_xlim(-0.7,4.7)

axes[4,1].set\_xlabel("Heart Disease [num]")

axes[4,1].set\_ylabel("Number of Patients")

axes[4,2].axis("off")

plt.show()

**#Converting categorial values into discrete values**

**#Note that feature ca is discrete but not categorical, so we don't convert it.**

df = table.copy()

dummies = pd.get\_dummies(df["cp"],prefix="cp")

df = df.join(dummies)

del df["cp"]

del df["cp\_4.0"]

df = df.rename(columns = {"cp\_1.0":"cp\_1","cp\_2.0":"cp\_2","cp\_3.0":"cp\_3"})

dummies = pd.get\_dummies(df["restecg"],prefix="recg")

df = df.join(dummies)

del df["restecg"]

del df["recg\_0.0"]

df = df.rename(columns = {"recg\_1.0":"recg\_1","recg\_2.0":"recg\_2"})

dummies = pd.get\_dummies(df["slope"],prefix="slope")

df = df.join(dummies)

del df["slope"]

del df["slope\_2.0"]

df = df.rename(columns = {"slope\_1.0":"slope\_1","slope\_3.0":"slope\_3"})

dummies = pd.get\_dummies(df["thal"],prefix="thal")

df = df.join(dummies)

del df["thal"]

del df["thal\_3.0"]

df = df.rename(columns = {"thal\_6.0":"thal\_6","thal\_7.0":"thal\_7"})

**#Replace response variable values and renaming it hd**

**#The num variable values of 1,2,3,4 are replaced with 1 in order to signify that the patient has heart disease**

df["num"].replace(to\_replace=[1,2,3,4],value=1,inplace=True)

df = df.rename(columns = {"num":"hd"})

new\_columns\_1 = ["age", "sex", "restbp", "chol", "fbs", "thalach", "exang", "oldpeak", "ca", "hd", "cp\_1", "cp\_2", "cp\_3", "recg\_1", "recg\_2", "slope\_1", "slope\_3", "thal\_6", "thal\_7"]

print ('\nNumber of patients in dataframe: %i, with disease: %i, without disease: %i\n' \ % (len(df.index),len(df[df.hd==1].index),len(df[df.hd==0].index)))

print (df.head())

print (df.describe())

**# Standardize the dataframe**

stdcols = ["age","restbp","chol","thalach","oldpeak","ca"]

stddf = df.copy()

stddf[stdcols] = stddf[stdcols].apply(lambda x: (x-x.mean())/x.std())

new\_columns\_2 = new\_columns\_1[:9] + new\_columns\_1[10:]

new\_columns\_2.insert(0,new\_columns\_1[9])

stddf = stddf.reindex(columns=new\_columns\_2)

**# Convert dataframe into lists for use by classifiers**

y = stddf["hd"]

X = stddf[new\_columns\_2[1:]].values

**# Training and testing data**

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

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.5, random\_state=1)

from sklearn.linear\_model import LogisticRegression

logmodel = LogisticRegression()

logmodel.fit(X\_train, y\_train)

predictions = logmodel.predict(X\_test)

from sklearn.metrics import classification\_report

classification\_report(y\_test, predictions)

from sklearn.metrics import confusion\_matrix

confusion\_matrix(y\_test, predictions)

from sklearn.metrics import accuracy\_score

accuracy\_score(y\_test, predictions)

print ('\nLogisticRegression coefficients:')

coeff = model.coef\_.tolist()[0]

for index in range(len(coeff)):

print ('%s : %8.5f' % (new\_columns\_2[index+1].rjust(9),coeff[index]))

print( 'Intercept : %f' %model.intercept\_)

**HeartDiseaseFrontEnd.html**

<**html**>  
  
<**head**>  
  
 <**h2**> <**center**> HEART DISEASE PREDICTOR </**center**> </**h2**>  
  
 <**script**>  
  
 function calc(form)  
 {  
 val00 = document.getElementById("nAge").value;  
 val01 = document.getElementById("nSex").value;  
 val02 = document.getElementById("nRestbp").value;  
 val03 = document.getElementById("nThalach").value;  
 val04 = document.getElementById("nCa").value;  
 val05 = document.getElementById("nCp").value;  
 val06 = document.getElementById("nSlope").value;  
 val07 = document.getElementById("nThal").value;  
  
 // Standardize the continuous features using z-score formula  
  
 let sage\_val = (val00 - 54.521) / 9.03;  
 let srbp\_val = (val02 - 131.7157) / 17.7478;  
 let sthalach\_val = (val03 - 149.3278) / 23.1211;  
 let sca\_val = (val04 - 0.6722) / 0.937;  
 // Disentangle the categorical features  
  
 let cp\_count = 0.0;  
 if (val05 === 1)  
 {  
 cp\_count = -2.10032;  
 } else if (val05 === 2)  
 {  
 cp\_count= -0.8;  
 } else if (val05 === 3)  
 {  
 cp\_count = -1.8;  
 }  
 let slope\_count = 0.0;  
 if (val06 === 0)  
 {  
 slope\_count = -1.16379;  
 } else if (val06 === 1)  
 {  
 slope\_count = -0.64688;  
 }  
 let thal\_count = 0.0;  
 if (val07 === 0)  
 {  
 thal\_count = -0.03066;  
 } else if (val07 === 1)  
 {  
 thal\_count = 1.40155;  
 }  
  
 // Compute log-odds, then probability  
  
 let prod = -0.03438 \* sage\_val + 1.54868 \* val01 + 0.43319 \* srbp\_val - 0.41727 \* sthalach\_val + 3.92866 \* sca\_val + cp\_count + slope\_count + thal\_count - 0.605239;  
 let result = 1.0 / (1.0 + Math.exp(-prod));  
 result \*= 100;  
 alert("Predicted value: " + result + "\n");  
  
 }  
 </**script**>  
  
</**head**>  
  
  
<**body**>  
 <**table**>  
 <**form name=myform action="" method="getElementById"**>  
  
 Age  
 <**input type="number" name="age" min="1" max="100" id="nAge"**>  
 <**br**/><**br**/>  
  
 Sex(0=female,1=male)  
  
 <**select name="sex" id="nSex"**>  
 <**option**> 0 </**option**>  
 <**option**> 1 </**option**>  
 </**select**>  
   
 <**br**/><**br**/>  
  
 Resting Blood Pressure(94-200mmHg)  
 <**input type="number" name="restbp" min="94" max="200" id="nRestbp"**>  
 <**br**/><**br**/>  
  
 Thalium Stress Test Maximum Heart Rate(71-202)  
 <**input type="number" name="nThalach1" min="71" max="202" id="nThalach"**>  
 <**br**/><**br**/>  
  
 Number of Major vessels colored by Fluoroscopy(0-3)  
 <**select name="color" id="nCa"**>  
 <**option**> 0 </**option**>  
 <**option**> 1 </**option**>  
 <**option**> 2 </**option**>  
 <**option**> 3 </**option**>  
 </**select**>  
 <**br**/><**br**/>  
  
 Chest Pain Type(1=typical angina,2=atypical angina,3=non-angina,4=asymptomatic angina)  
 <**select name="pain" id="nCp"**>  
 <**option**> 1 </**option**>  
 <**option**> 2 </**option**>  
 <**option**> 3 </**option**>  
 </**select**>  
 <**br**><**br**/>  
  
 Peak Excercise ST Segment(0=flat or downsloping,1=unsloping)  
 <**select name="segment" id="nSlope"**>  
 <**option**> 0 </**option**>  
 <**option**> 1 </**option**>  
 </**select**>  
 <**br**/><**br**/>  
  
 Thalium Test(0=normal or fixed defect,1=reversible defect)  
 <**select name="test" id="nThal"**>  
 <**option**> 0 </**option**>  
 <**option**> 1 </**option**>  
 </**select**>  
 <**br**/><**br**/>  
  
 <**input type="button" name="button" Value = "Predict" onclick="**calc(this.form)**"** />  
 </**form**>  
 </**table**>  
 </**body**>  
</**html**>

**Chapter 6**

**TESTING**

## 6.1 System Testing

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, subassemblies, assemblies and/or a finished product It is the process of exercising software with the intent of ensuring that the

Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

### **6.1.1 Unit Testing**

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results. Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

* **Test strategy and approach**

Field testing will be performed manually and functional tests will be written in detail.

* **Test objectives** 
  + All field entries must work properly.
  + Pages must be activated from the identified link.
  + The entry screen, messages and responses must not be delayed.

* **Features to be tested** 
  + Verify that the entries are of the correct format
  + No duplicate entries should be allowed
  + All links should take the user to the correct page.

### **6.1.2 Black Box Testing**

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box you cannot “see” into it. The test provides inputs and responds to outputs without considering how the software works.

## 6.2 Functional Test Cases

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name** | **Description** | **Expected output** | **Actual output** | **Result** |
| Load dataset | Dataset to be loaded from UCI repository | Selected files to be downloaded to project directory | Selected files to be downloaded to project directory | Success |
| File type conversion | Converting ssv file to csv | Converting ssv file to csv | Converting ssv file to csv | Success |
| Histograms generation | Plotting histograms | Relation between number of patients and various attributes | Relation between number of patients and various attributes | Success |
| Value conversion | Converting categorical values to discrete values | Attributes consisting of categorical values converted to discrete values | Attributes consisting of categorical values converted to discrete values | Success |
| Min-max normalization | Make every data point have the same scale | Standardization of data | Transformed values lie between limited interval [0,1] | Failure |
| Z-score | Make every data point have the same scale | Standardization of data | Transformed values lie between [ - infinity, infinity] | Success |

## 6.3 Screenshots

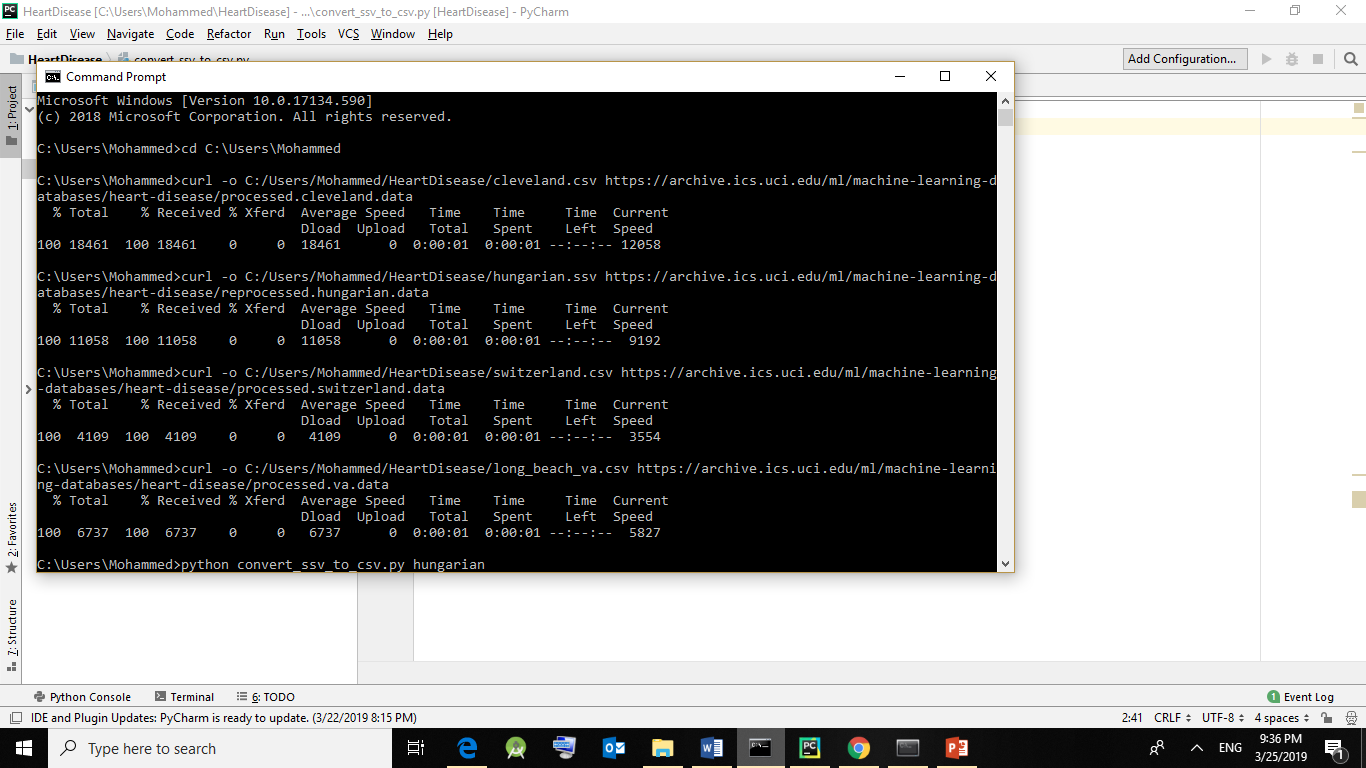


Fig 5.4(i) Reading datasets from UCI repository & converting ssv file to csv

* The first step of the project is to download the datasets from UCI repository using curl-o command which are saved into our project directory.

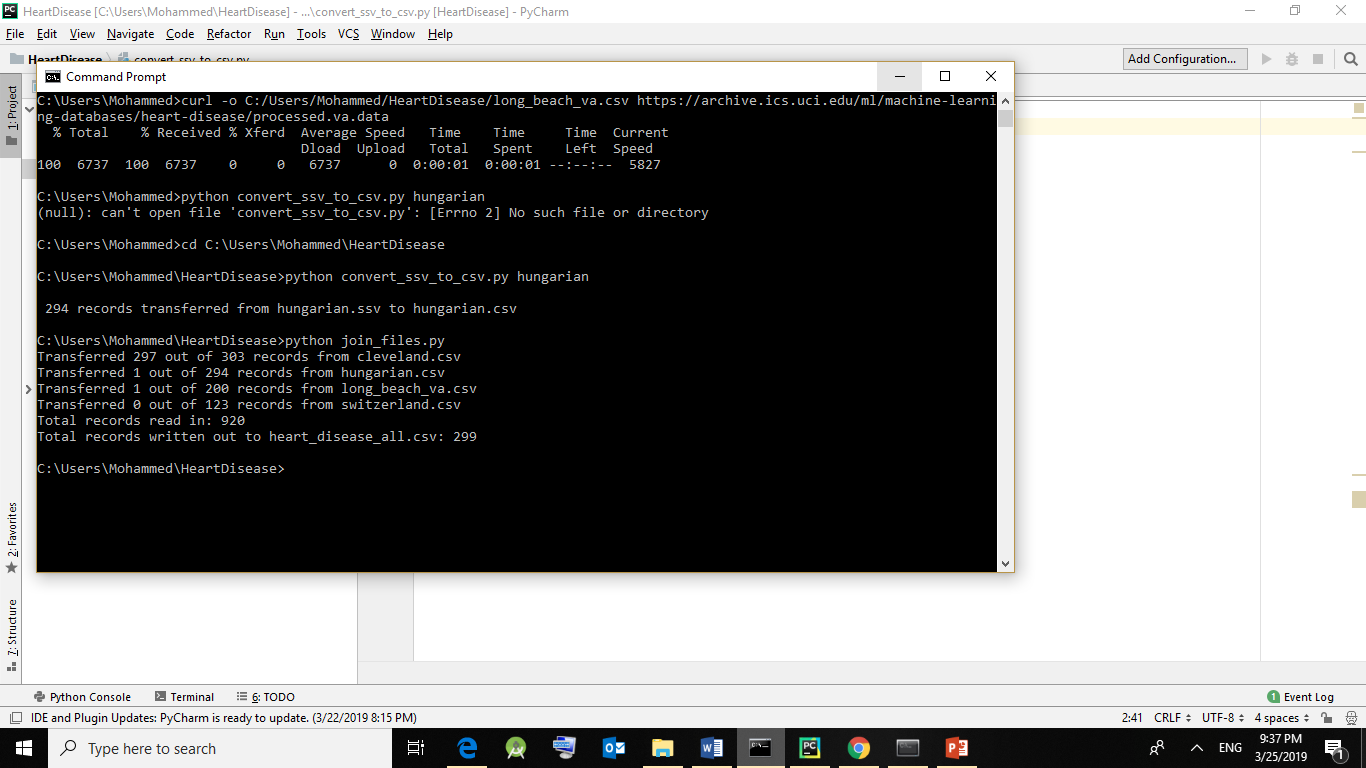


Fig 5.4(ii) Converting ssv file to csv and combining all csv files

* Using the file convert\_ssv\_to\_csv, we convert Hungarian.ssv to Hungarian.csv so that all datasets have uniform file type.
* Using join\_files, we join the four datasets (Cleveland.csv, Hungarian.csv, long\_beach.csv, Switzerland.csv) and transfer the records to one single file i.e. heart\_disease\_all.csv.
* Join\_files transforms only those records that do not have any missing values in them and removes all others.

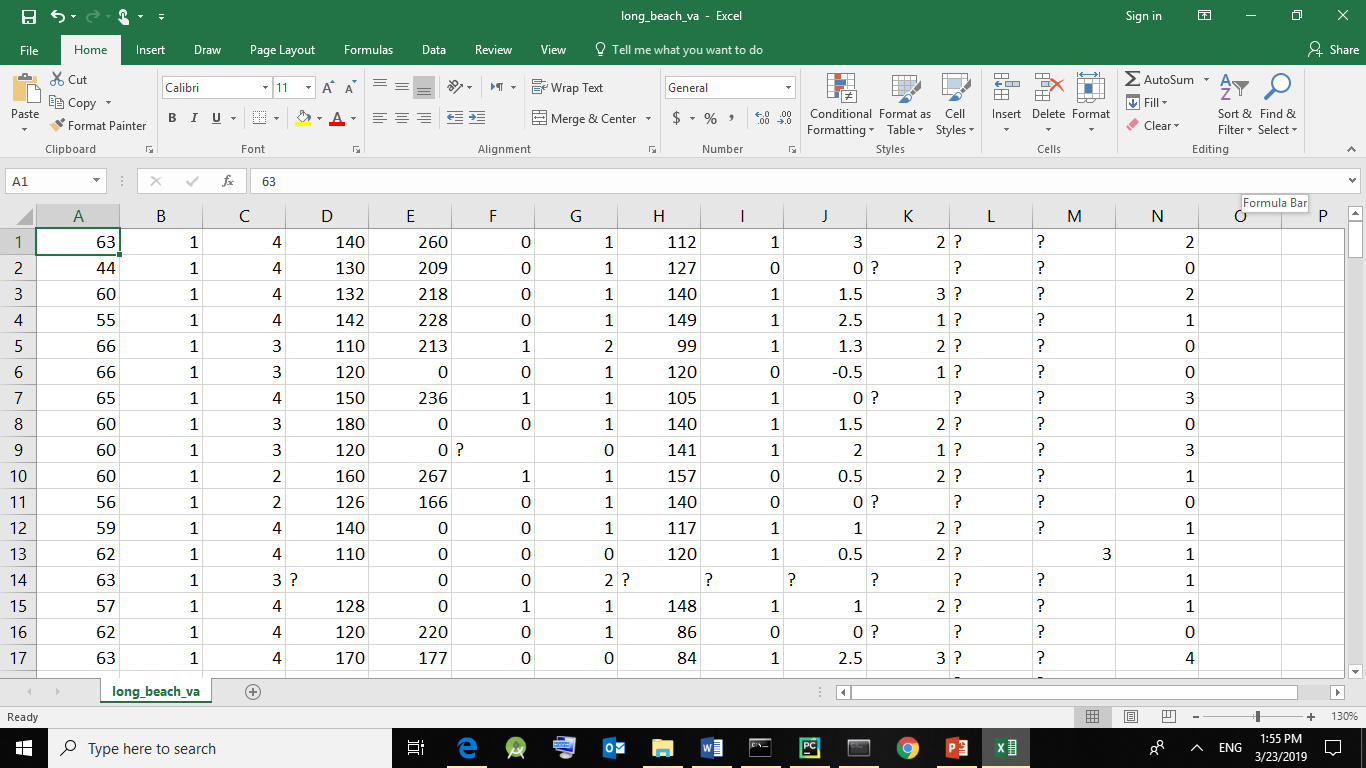


Fig 5.4(iii) Dataset consisting of missing values

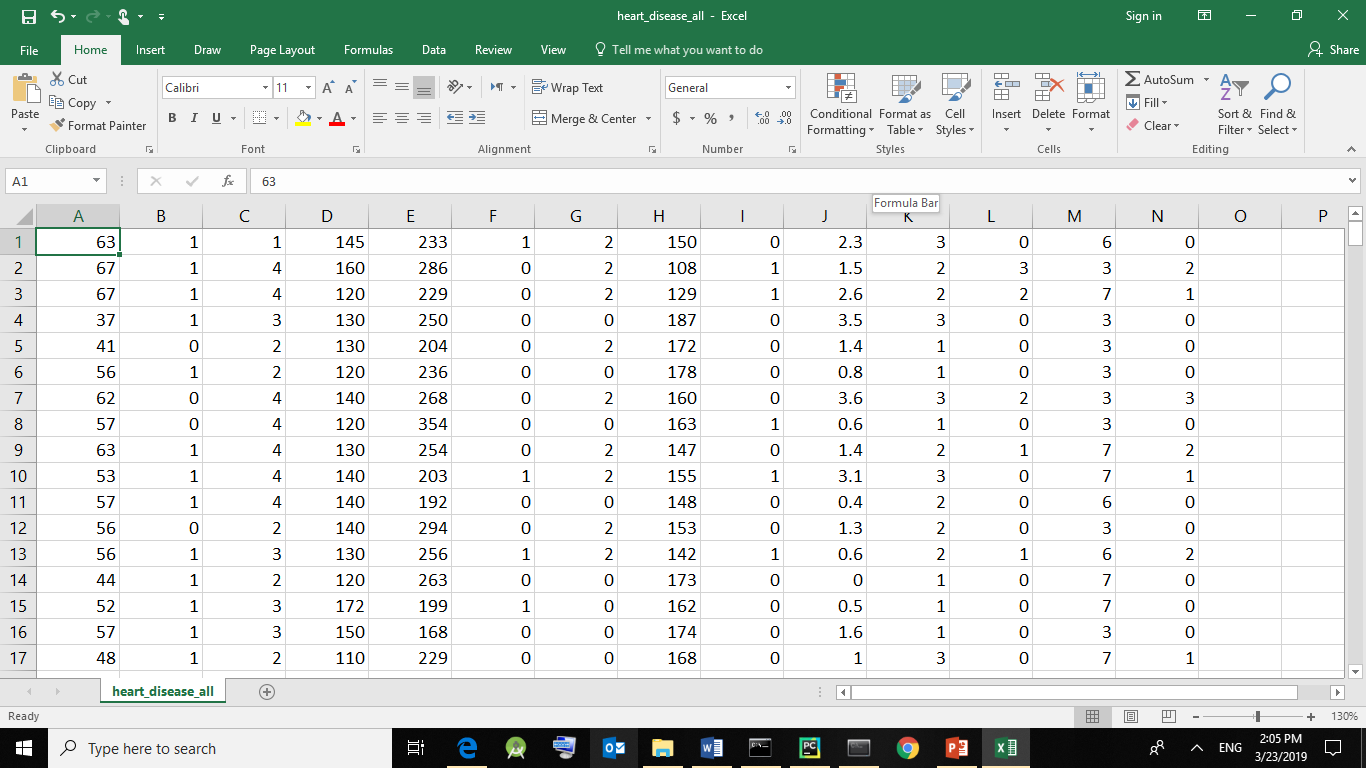


Fig 5.4(iv) After data cleaning

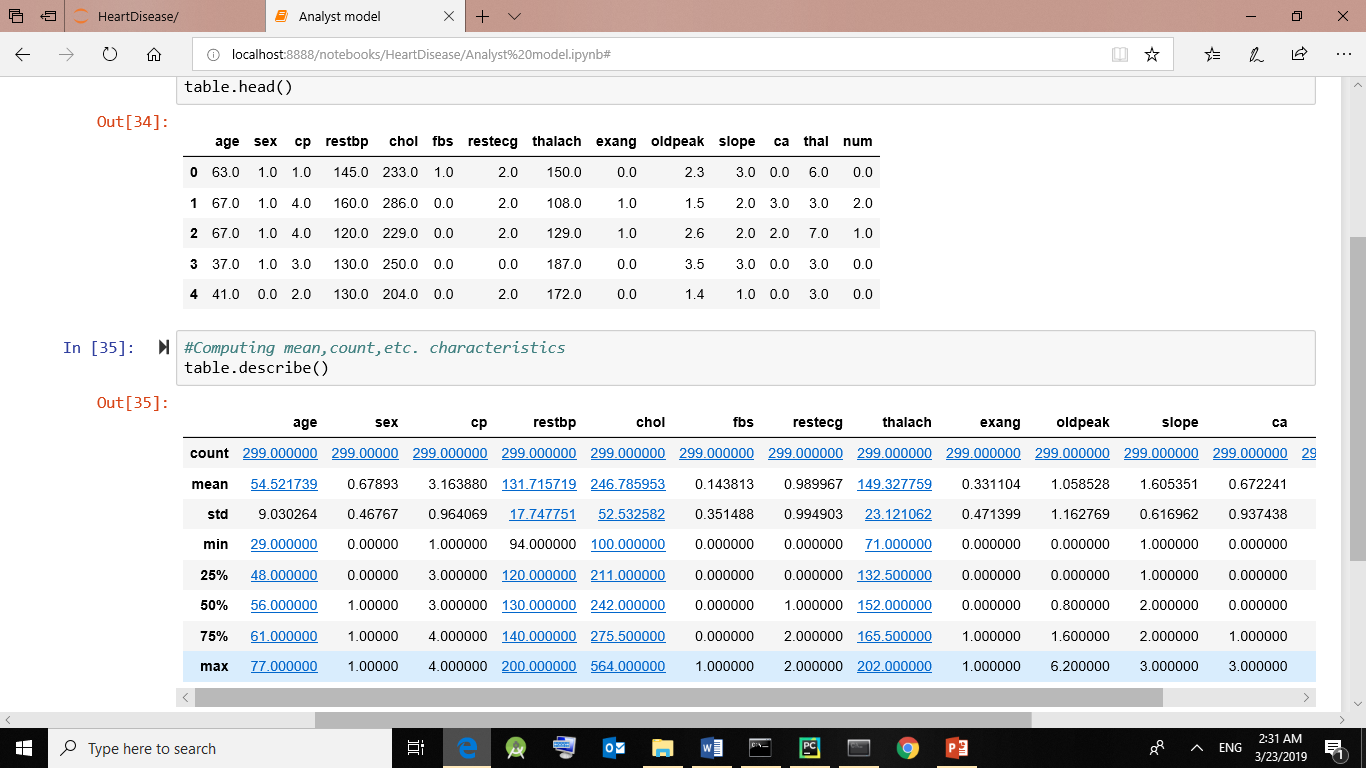


Fig 5.4(v) Naming the columns and computing the characteristics

* The first step is to load the dataset and name all the columns

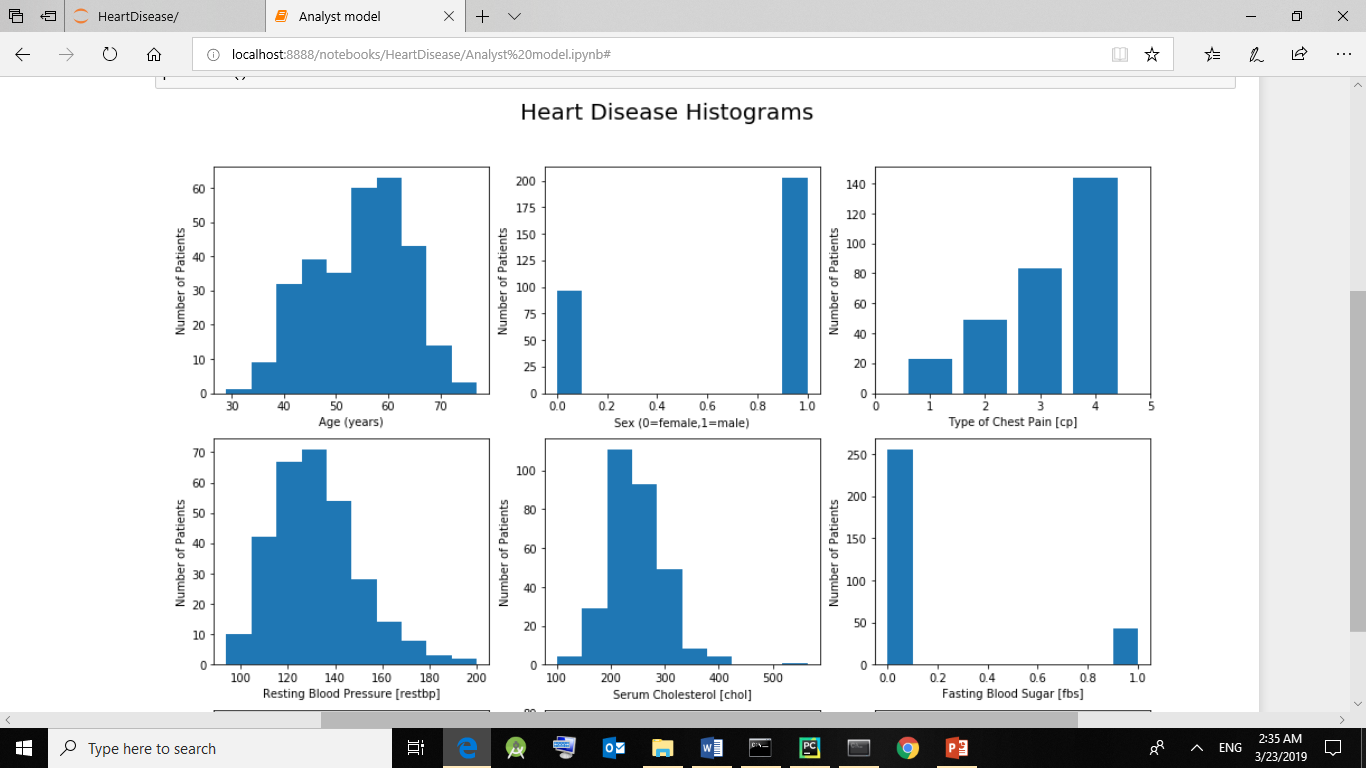


Fig 5.4(vi) Generating histograms of the given dataset

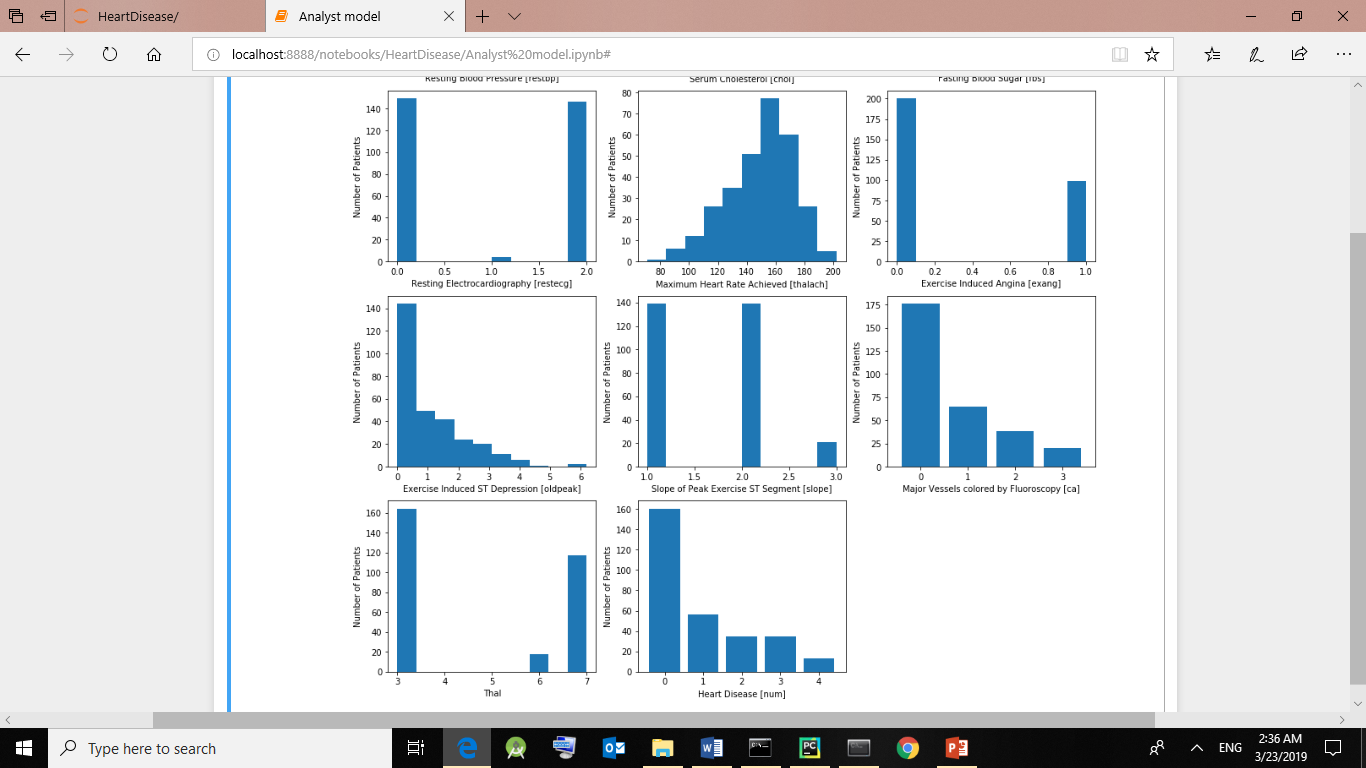


Fig 5.4(vii) Dataset consisting of missing values

* Next, we generate histograms for graphical analysis of the data. The histograms are drawn to show the relationship between the number of patients and each attribute of the dataset.

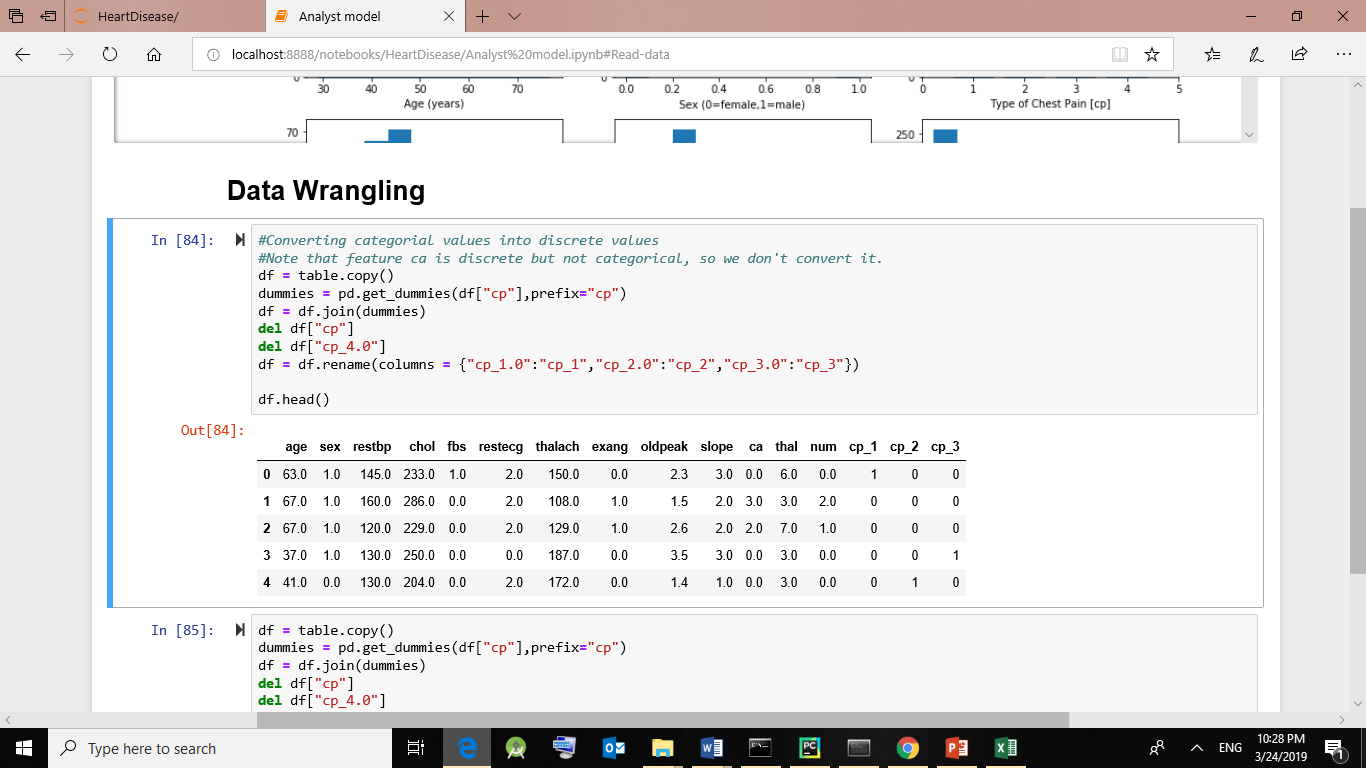


Fig 5.4(viii) Converting categorical value for one attribute (cp) to discrete value

* Data wrangling basically consists of converting categorical values into discrete values.
* The main distinction is between continuous variables (such as income, age and blood pressure) and discrete variables (such as sex or race). Discrete variables refer to more than two possible choices and are coded using dummy variables i. e separate explanatory variables taking the value 0 or 1 are created for each possible value of the discrete variable, with a 1 meaning "variable does have the given value" and a 0 meaning "variable does not have that value".
* Example:

Here, cp (chest pain type) has four values each meaning the following –

Value 1: typical angina   
Value 2: atypical angina   
Value 3: non-anginal pain   
Value 4: asymptomatic

Similarly, the attributes restecg, slope, thal are also converted.

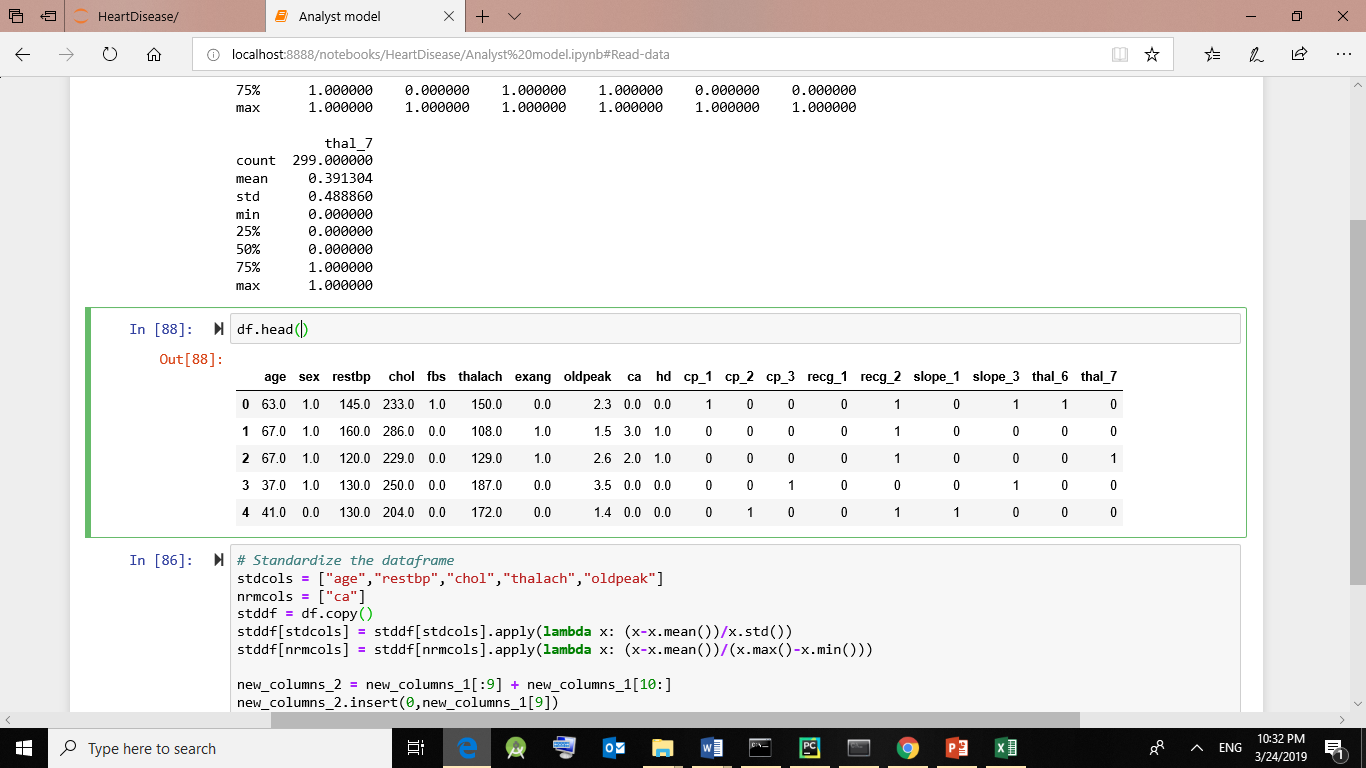


Fig 5.4(ix) Converting categorical value to discrete value of entire dataset

Restecg-  resting electrocardiographic results   
Value 0: normal   
Value 1: having ST-T wave abnormality

Value 2: showing probable or definite left ventricular hypertrophy by Estes' criteria

Slope -  the slope of the peak exercise ST segment   
-- Value 1: upsloping   
-- Value 2: flat   
-- Value 3: downsloping

Thal

3 = normal

6 = fixed defect

7 = reversable defect

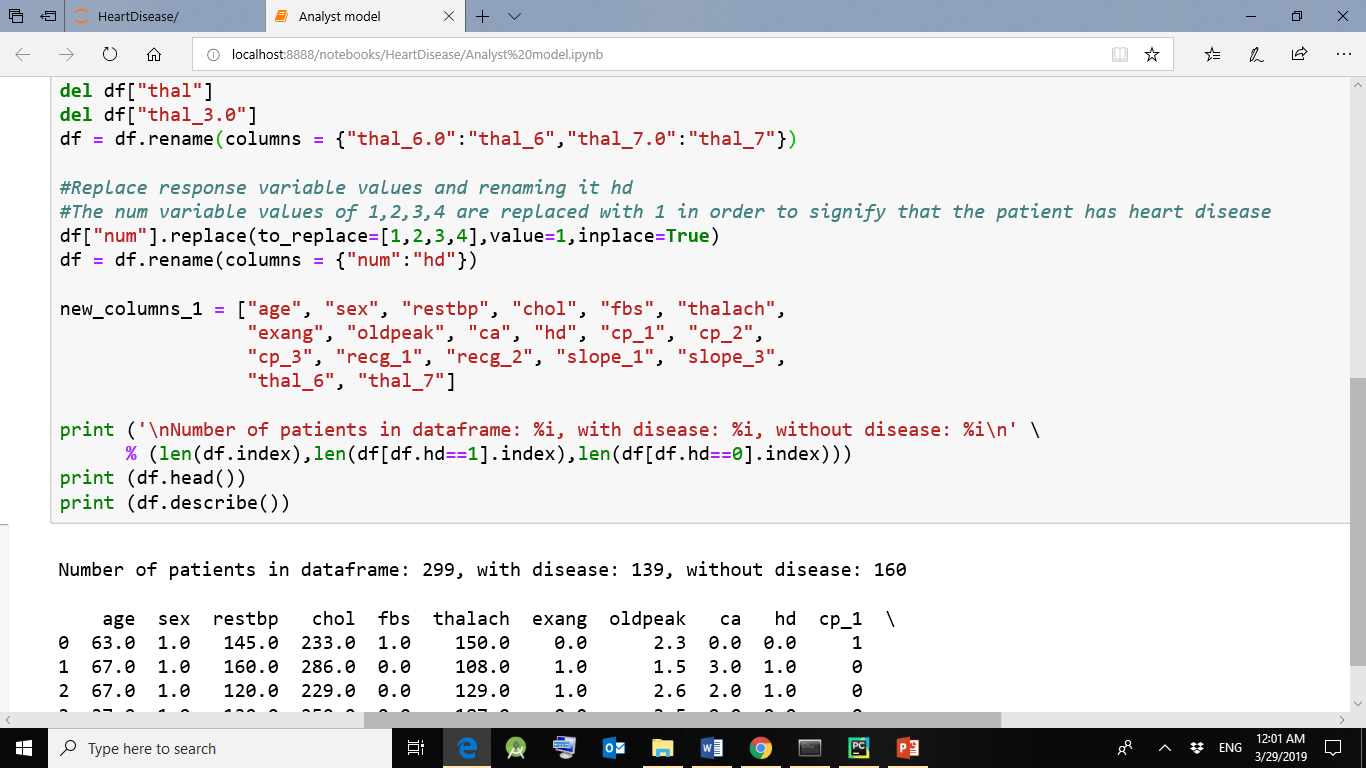


Fig 5.4(x) Converting num values to binary

* In our dataset, the “num” attribute that is the diagnosis of heart disease has the values ranging from 0-4 where 0 depicts that there is no heart disease diagnosed and 1-4 depicting otherwise.
* So, we replace the values 1,2,3,4 as 1 in general, signifying that patient has heart disease.

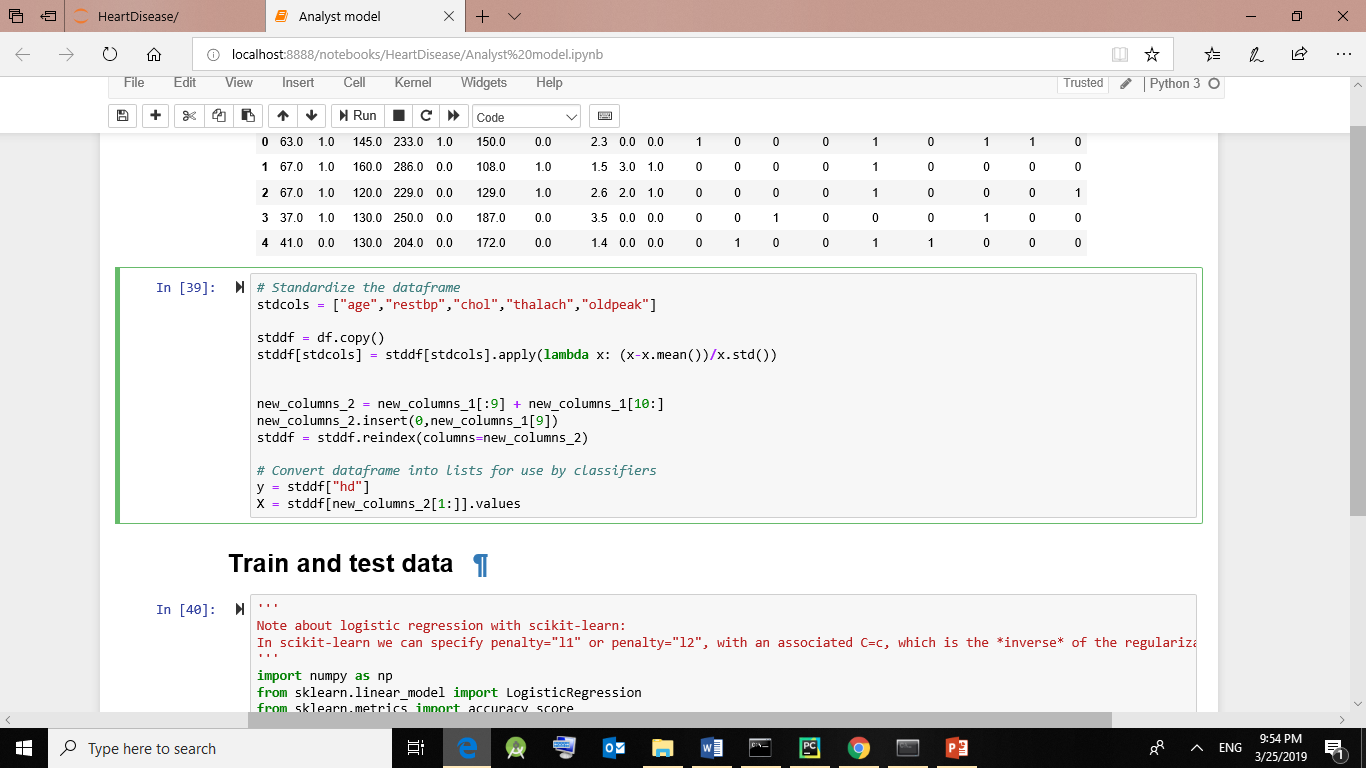


Fig 5.4(xi) Normalization/Standardization of data using z-score method

* Using z – score normalization,

z = (x – μ) / σ

where x – sample mean

μ – population mean

σ – standard deviation

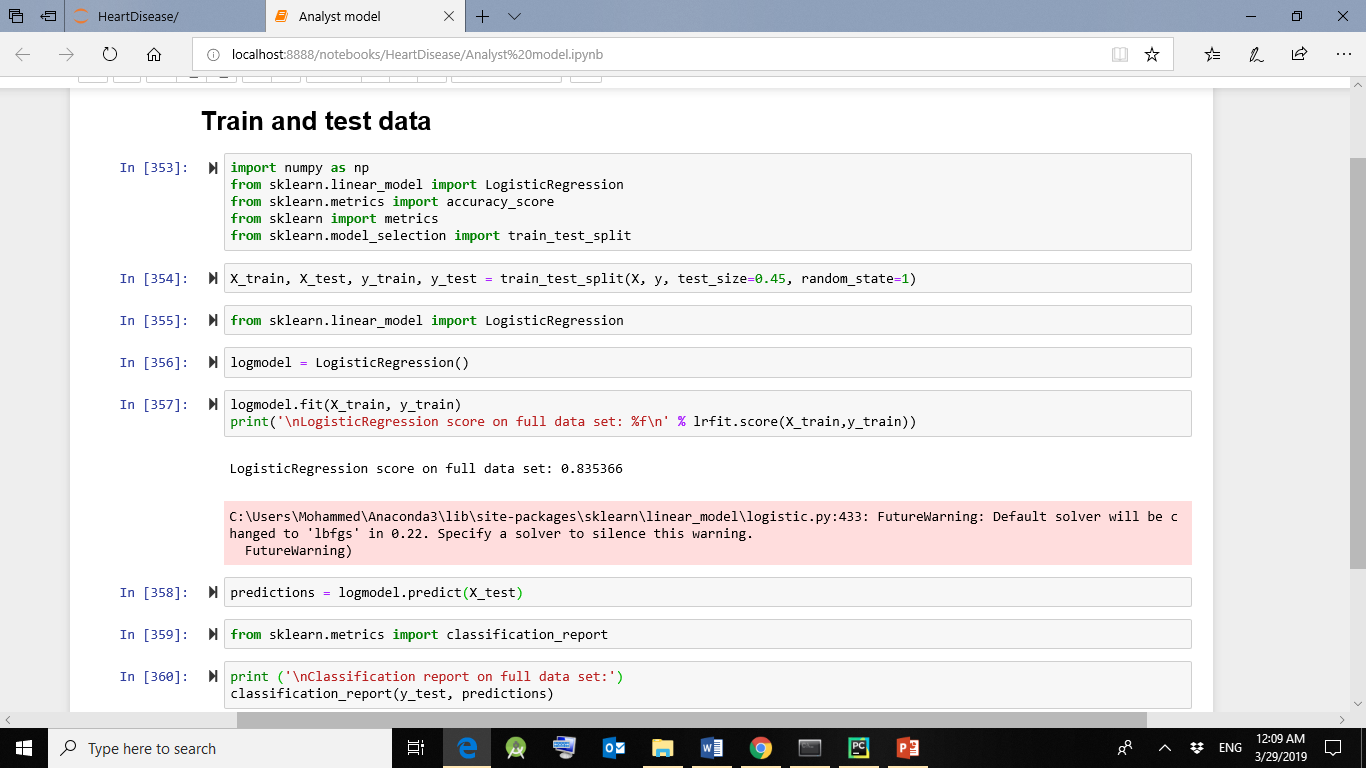


Fig 5.4(xii) Training and testing the model

* First, we import all necessary libraries.
* Then, the training and testing data is divided into 45-55% and the data is fit into the logistic regression model.
* The accuracy, regression coefficients and intercept are calculated.
* A regression coefficient describes the size and direction of the relationship between a predictor and the response variable. Coefficients are the numbers by which the values of the term are multiplied in a regression equation.

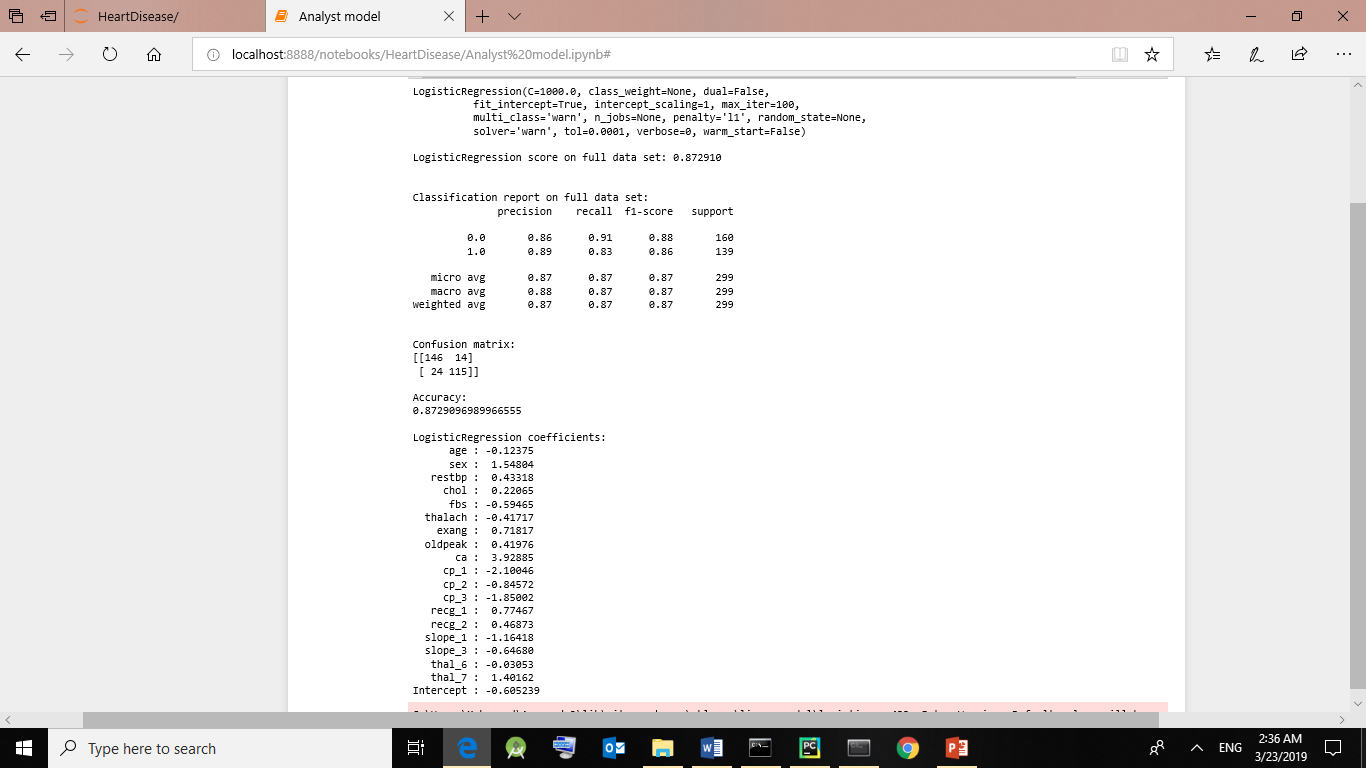


Fig 5.4(xiii) Applying logistic regression to the dataset and calculating its accuracy and respective coefficients/slope

Prediction formula-

Y=w[0] + w[1] \* x[1] + . . . + w[p] \* x[p] + b > 0

where w – slope (or coefficient),

b – offset ( or intercept),

w and b are learned parameters

Y – prediction (or decision)

To obtain predicted value between 0-1, we use the sigmoid function -

H=1/(1+e-y)

Where H – output value,

Y – predicted value

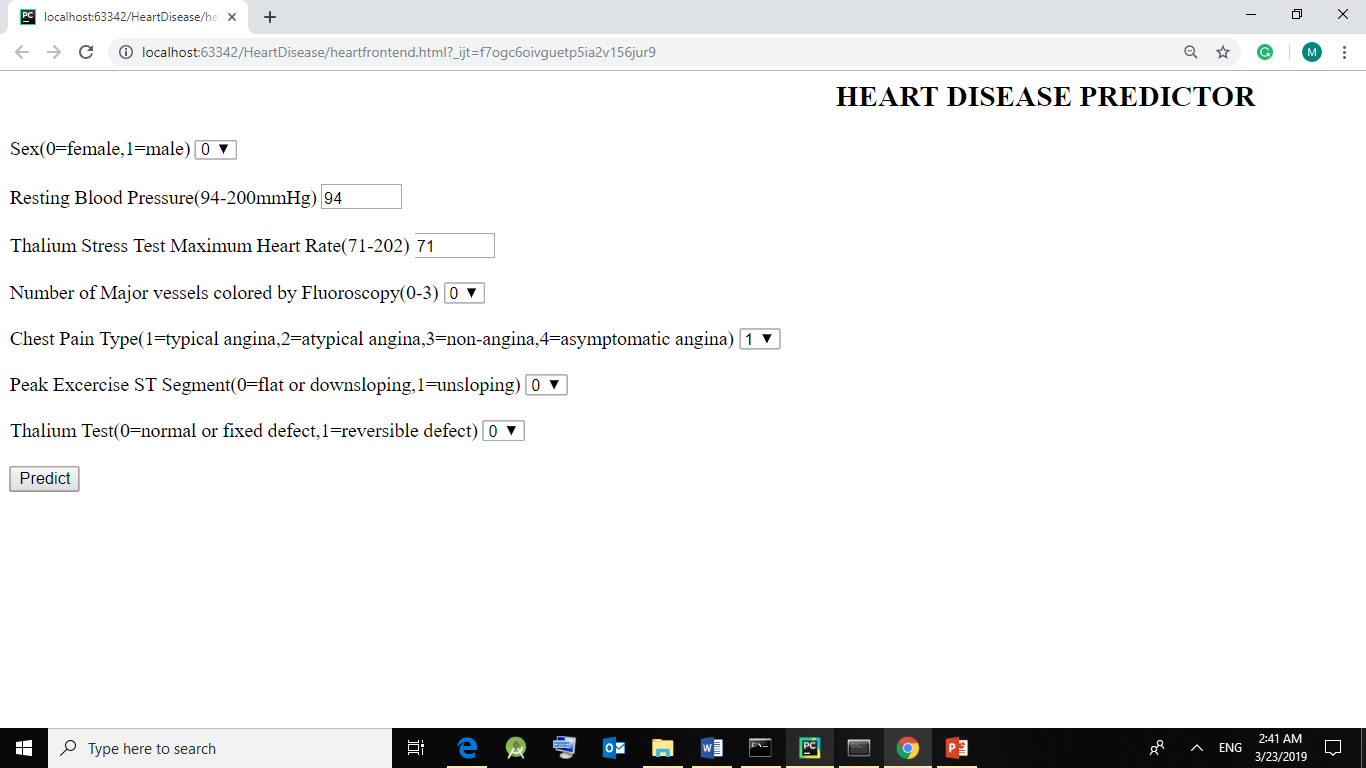


Fig 5.4(xiv) Form to take input values from user to calculate the risk

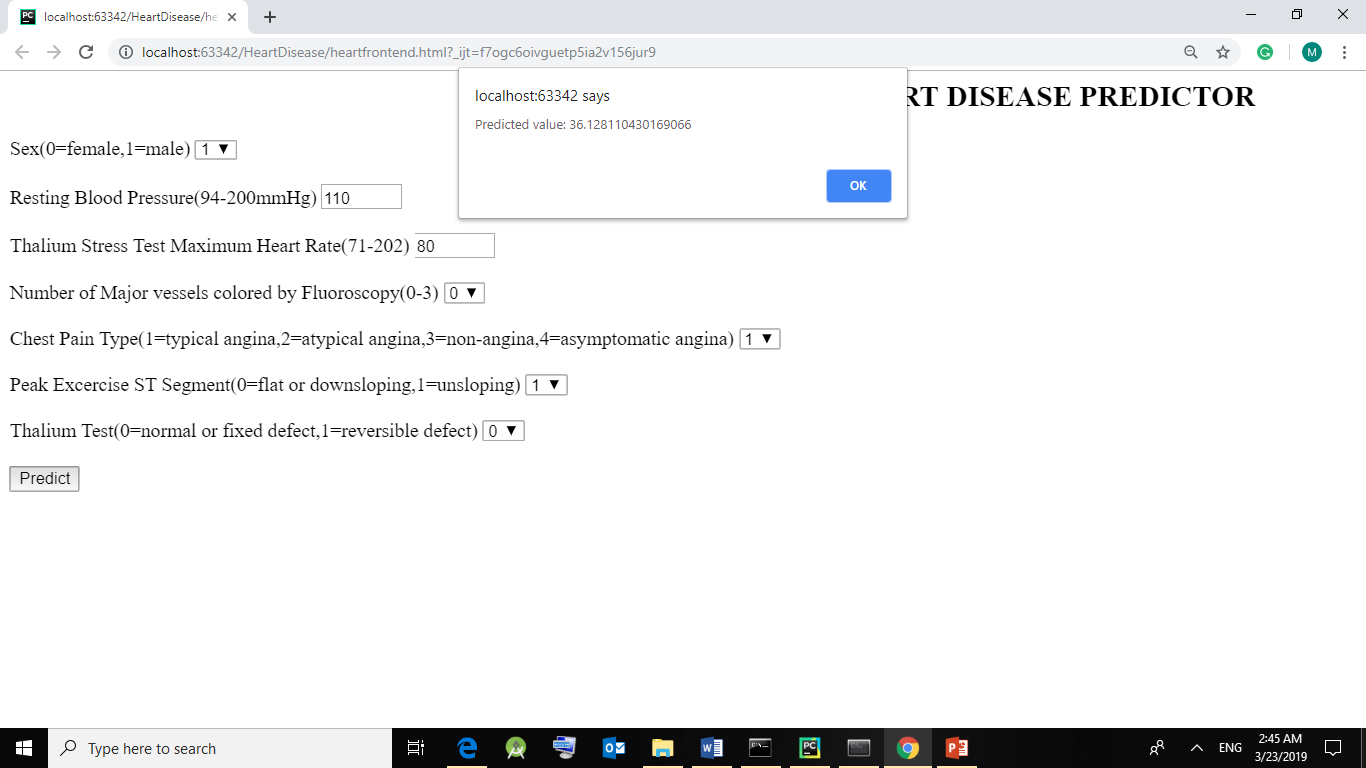


Fig 5.4(xv) Form to take input values from user to calculate the risk

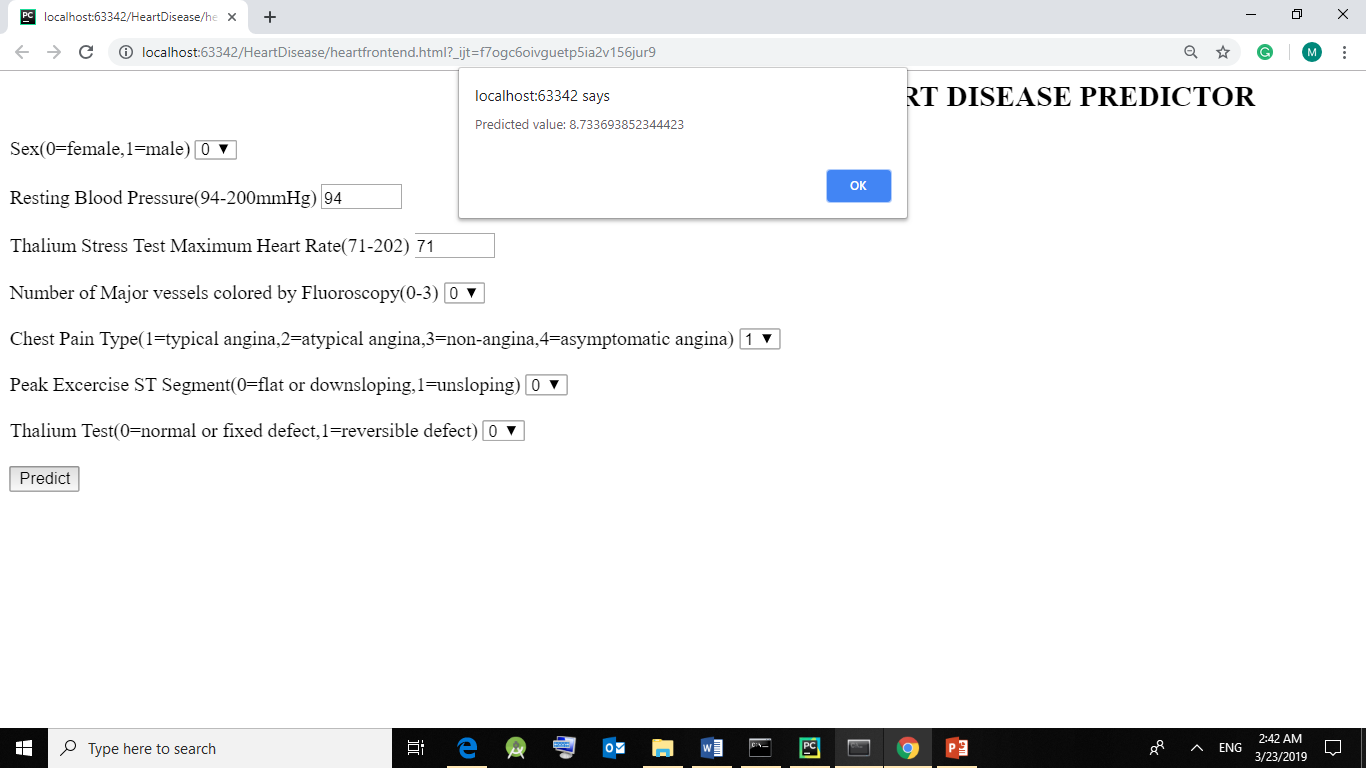
****

Fig 5.4(xvi) Form to take input values from user to calculate the risk

* Finally, we use the regression coefficients and intercept to calculate the final output using the following formula in our HTML.
* Here, we take input from the user which are calculated against the coefficients and intercept using prediction formula.

**Chapter 7**

**CONCLUSION**

## 7.1 Summary

This system proposes confidential scheme for predicting heart disease using, Logistic Regression. As identified through survey, it is a need to have combinational approach to

increase the accuracy of prediction for heart disease. The system has been implemented with the accuracy of 83.67% on the dataset taken from UCI. The current system covers only the cancer diseases, the plan is to include general disease of higher fatality, so that early prediction and treatment could be done, and the fatality rate of deadly diseases decreases, with the economic benefit in long sight as well.

## 7.2 Scope for Future Work

It is not possible to develop a system that makes all the requirements of the user, user requirements keep changing as the system is being used. Some of the future enhancements that can be done to this system are:

* Increase the accuracy by increasing the training set data.
* Increasing dataset attributes for more types of diseases.
* Adding more features or information if the user desires.
* Fixing the bugs/ problems which user encounters.
* Better scaling of the application.

**References**

1. Ashish Kumar Sen, Shamsher Bahadur Patel, Dr. D. P. Shukla, “A Data Mining Technique for Prediction of Coronary Heart Disease Using Neuro-Fuzzy Integrated” Approach Two Level”, International Journal Of Engineering And Computer Science ISSN:2319-7242 Volume 2 Issue 9 Sept., 2013 Page No. 2663-2671.
2. Syed Umar Amin, Kavita Agarwal, Rizwan Beg,, “ Genetic neural network based data mining in prediction of heart disease using risk factors”, Information and communication technologies (ICT), 2013 conference on information and communication technologies.
3. Jyoti Soni, Ujma Ansari, Dipesh Sharma, Sunita Soni: Predictive Data Mining for Medical Diagnosis: An Overview of Heart Disease Prediction”, International Journal of Computer Applications (0975 – 8887), Volume 17– No.8, March 2011.
4. Divya Tomar and Sonali Agarwal, “ A survey on Data Mining approaches for Healthcare”, International Journal of Bio-Science and Bio-Technology, Vol.5, No.5 (2013), pp. 241-266.
5. M. Anbarasi, E. Anupriya and N.CH.S.N. Iyenga, “Enhanced Prediction of Heart Disease with Feature Subset Selection using Genetic Algorithm”, International Journal of Engineering Science and Technology, Vol. 2, No. 10, pp. 5370 - 5376, 2010.
6. Chaitrali S. Dangare and Sulabha S. Apte, “Improved Study of Heart Disease Prediction System using Data Mining Classification Techniques”, International Journal of Computer Applications, Vol. 47, No. 10, pp. 0975 – 888, 2012.
7. P. K. Anooj, “Clinical decision support system: Risk level prediction of heart disease using weighted fuzzy rules”, Journal of King Saud University Computer and Information Sciences, Vol. 11, pp. 309 - 314, 2011.
8. Mai Shouman, Tim Turner and Rob Stocker, “Integrating Decision Tree and Logistic Regression with Different Initial Centroid Selection Methods in the Diagnosis of Heart Disease Patients”, Proceedings of the International Conference on Data Mining, 2012.

**APPENDIX A**

**Software and Hardware Requirements**

* **Software Requirements**
* **OS:** Windows, Linux, Mac
* **Coding language:** Python 3.7
* **Documentation:** MS Office 2013
* **IDE:** PyCharm
* **Libraries:** pandas, NumPy, matplotlib, sklearn
* Anaconda 3.7.1

**Web Browser:** Microsoft Edge, Google Chrome, Mozilla Firefox

* **Hardware Requirements**

**Hardware:** Pentium dual core

**Speed:** 2.2 GHz

**RAM:** 8 GB

**Hard disk:** 20GB

**APPENDIX B**

**Technologies Used**

* **Python**

Python is an interpreted, high-level, general-purpose programming language. Created by Guido van Rossum and first released in 1991, Python has a design philosophy that emphasizes code readability, notably using significant whitespace. It provides constructs that enable clear programming on both small and large scales. Van Rossum led the language community until stepping down as leader in July 2018. Python features a dynamic type system and automatic memory management.

Python is a multi-paradigm programming language. Object-oriented programming and structured programming are fully supported, and many of its features support functional programming and aspect-oriented programming (including by metaprogramming and metaobjects). Many other paradigms are supported via extensions, including design by contract and logic programming.

Python interpreters are available for many operating systems. CPython, the reference implementation of Python, is open source software and has a community-based development model, as do nearly all of Python's other implementations. Python and CPython are managed by the non-profit Python Software Foundation.

* **PyCharm**

**PyCharm** is an [integrated development environment](https://en.wikipedia.org/wiki/Integrated_development_environment) (IDE) used in [computer programming](https://en.wikipedia.org/wiki/Computer_programming), specifically for the [Python](https://en.wikipedia.org/wiki/Python_(programming_language)) language. It is developed by the Czech company [JetBrains](https://en.wikipedia.org/wiki/JetBrains) It provides code analysis, a graphical debugger, an integrated unit tester, integration with [version control systems](https://en.wikipedia.org/wiki/Revision_control) (VCSes), and supports web development with [Django](https://en.wikipedia.org/wiki/Django_(web_framework)).

PyCharm is [cross-platform](https://en.wikipedia.org/wiki/Cross-platform), with [Windows](https://en.wikipedia.org/wiki/Windows), [macOS](https://en.wikipedia.org/wiki/MacOS) and [Linux](https://en.wikipedia.org/wiki/Linux) versions. The Community Edition is released under the [Apache License](https://en.wikipedia.org/wiki/Apache_License), and there is also Professional Edition with extra features, released under a [proprietary license](https://en.wikipedia.org/wiki/Proprietary_software).

Following are its features-

* + Coding assistance and [analysis](https://en.wikipedia.org/wiki/Code_analysis), with [code completion](https://en.wikipedia.org/wiki/Autocomplete), syntax and error highlighting, linter integration, and quick fixes
  + Project and code navigation: specialized project views, file structure views and quick jumping between files, classes, methods and usages
  + Python [refactoring](https://en.wikipedia.org/wiki/Refactoring): including rename, extract method, introduce variable, introduce constant, pull up, push down and others
  + Support for web frameworks: [Django](https://en.wikipedia.org/wiki/Django_(web_framework)), [web2py](https://en.wikipedia.org/wiki/Web2py) and [Flask](https://en.wikipedia.org/wiki/Flask_(web_framework))
  + Integrated Python [debugger](https://en.wikipedia.org/wiki/Debugger)
  + Integrated [unit testing](https://en.wikipedia.org/wiki/Unit_testing), with line-by-line [code coverage](https://en.wikipedia.org/wiki/Code_coverage)
  + [Google App Engine](https://en.wikipedia.org/wiki/Google_App_Engine) Python development
  + Version control integration: unified user interface for [Mercurial](https://en.wikipedia.org/wiki/Mercurial), [Git](https://en.wikipedia.org/wiki/Git_(software)), [Subversion](https://en.wikipedia.org/wiki/Apache_Subversion), [Perforce](https://en.wikipedia.org/wiki/Perforce) and [CVS](https://en.wikipedia.org/wiki/Concurrent_Versions_System) with changelists and merge

It competes mainly with a number of other Python-oriented IDEs, including [Eclipse](https://en.wikipedia.org/wiki/Eclipse_(software))'s [PyDev](https://en.wikipedia.org/wiki/PyDev" \o "PyDev), and the more broadly focused Komodo IDE.

* **Anaconda**

Anaconda is a free and open-source distribution of the [Python](https://en.wikipedia.org/wiki/Python_(programming_language)) and [R](https://en.wikipedia.org/wiki/R_(programming_language)) programming languages for [scientific computing](https://en.wikipedia.org/wiki/Scientific_computing) ([data science](https://en.wikipedia.org/wiki/Data_science), [machine learning](https://en.wikipedia.org/wiki/Machine_learning) applications, large-scale data processing, [predictive analytics](https://en.wikipedia.org/wiki/Predictive_analytics), etc.), that aims to simplify [package management](https://en.wikipedia.org/wiki/Package_management) and deployment. Package versions are managed by the [package management system](https://en.wikipedia.org/wiki/Package_manager) *[conda](https://en.wikipedia.org/wiki/Conda_(package_manager)" \o "Conda (package manager))*. The Anaconda distribution is used by over 12 million users and includes more than 1400 popular data-science packages suitable for Windows, Linux, and MacOS.

**Anaconda Navigator**

Anaconda Navigator is a desktop [graphical user interface (GUI)](https://en.wikipedia.org/wiki/Graphical_user_interface) included in Anaconda distribution that allows users to launch applications and manage conda packages, environments and channels without using [command-line commands](https://en.wikipedia.org/wiki/Command-line_interface). Navigator can search for packages on Anaconda Cloud or in a local Anaconda Repository, install them in an environment, run the packages and update them. It is available for [Windows](https://en.wikipedia.org/wiki/Windows), [macOS](https://en.wikipedia.org/wiki/MacOS) and [Linux](https://en.wikipedia.org/wiki/Linux).

The following applications are available by default in Navigator:

* [JupyterLab](https://en.wikipedia.org/wiki/Project_Jupyter#Jupyter_Lab)
* [Jupyter Notebook](https://en.wikipedia.org/wiki/Project_Jupyter#Jupyter_Notebook)
* [QtConsole](https://qtconsole.readthedocs.io/en/latest/)
* [Spyder](https://en.wikipedia.org/wiki/Spyder_(software))
* [Glueviz](http://glueviz.org/)
* [Orange](https://en.wikipedia.org/wiki/Orange_(software))
* [Rstudio](https://en.wikipedia.org/wiki/Rstudio)
* [Visual Studio Code](https://en.wikipedia.org/wiki/Visual_Studio_Code)

### **Conda**

Conda is an [open source](https://en.wikipedia.org/wiki/Open-source_software), [cross-platform](https://en.wikipedia.org/wiki/Cross-platform), language-agnostic [package manager](https://en.wikipedia.org/wiki/Package_manager) and environment management system that installs, runs, and updates packages and their dependencies. It was created for Python programs, but it can package and distribute software for any language (e.g., [R](https://en.wikipedia.org/wiki/R_(programming_language))), including multi-language projects. The Conda package and environment manager is included in all versions of Anaconda, Miniconda, and Anaconda Repository.

### **Jupyter Notebook**

Jupyter [Notebook](https://en.wikipedia.org/wiki/Notebook_interface) (formerly IPython Notebooks) is a [web-based interactive](https://en.wikipedia.org/wiki/Rich_Internet_application) computational environment for creating Jupyter notebooks documents. The "notebook" term can colloquially make reference to many different entities, mainly the Jupyter web application, Jupyter Python web server, or Jupyter document format depending on context. A Jupyter Notebook document is a [JSON](https://en.wikipedia.org/wiki/JSON) document, following a versioned schema, and containing an ordered list of input/output cells which can contain code, text (using [Markdown](https://en.wikipedia.org/wiki/Markdown)), mathematics, plots and rich media, usually ending with the ".ipynb" extension.

A Jupyter Notebook can be converted to a number of [open standard](https://en.wikipedia.org/wiki/Open_standard) output formats ([HTML](https://en.wikipedia.org/wiki/HTML), [presentation slides](https://en.wikipedia.org/wiki/Presentation_slide), [LaTeX](https://en.wikipedia.org/wiki/LaTeX), [PDF](https://en.wikipedia.org/wiki/PDF), [ReStructuredText](https://en.wikipedia.org/wiki/ReStructuredText" \o "ReStructuredText), [Markdown](https://en.wikipedia.org/wiki/Markdown), [Python](https://en.wikipedia.org/wiki/Python_(programming_language))) through "Download As" in the web interface, via the [nbconvert](https://nbconvert.readthedocs.io/) library or "jupyter nbconvert" command line interface in a shell.

A Jupyter kernel is a program responsible for handling various types of request (code execution, code completions, inspection), and providing a reply. Jupyter Notebook can connect to many kernels to allow programming in many languages. By default Jupyter Notebook ships with the IPython kernel. As of the 2.3 release(October 2014), there are currently 49 Jupyter-compatible kernels for as many programming languages, including [Python](https://en.wikipedia.org/wiki/Python_(programming_language)), [R](https://en.wikipedia.org/wiki/R_(programming_language)), [Julia](https://en.wikipedia.org/wiki/Julia_(programming_language)) and [Haskell](https://en.wikipedia.org/wiki/Haskell_(programming_language)).

**Libraries**

* **Pandas**

Pandas is a [software library](https://en.wikipedia.org/wiki/Software_library) written for the [Python programming language](https://en.wikipedia.org/wiki/Python_(programming_language)) for data manipulation and analysis. In particular, it offers data structures and operations for manipulating numerical tables and [time series](https://en.wikipedia.org/wiki/Time_series). It is [free software](https://en.wikipedia.org/wiki/Free_software) released under the [three-clause BSD license](https://en.wikipedia.org/wiki/3-clause_BSD_license). The name is derived from the term "[panel data](https://en.wikipedia.org/wiki/Panel_data)", an [econometrics](https://en.wikipedia.org/wiki/Econometrics) term for data sets that include observations over multiple time periods for the same individuals.

Following are the library features-

* DataFrame object for data manipulation with integrated indexing.
* Tools for reading and writing data between in-memory data structures and different file formats.
* Data alignment and integrated handling of missing data.
* Reshaping and pivoting of data sets.
* Label-based slicing, fancy indexing, and subsetting of large data sets.
* Data structure column insertion and deletion.
* Group by engine allowing split-apply-combine operations on data sets.
* Data set merging and joining.
* Hierarchical axis indexing to work with high-dimensional data in a lower-dimensional data structure.
* Time series-functionality: Date range generation and frequency conversion, moving window statistics, moving window linear regressions, date shifting and lagging.
* Provides data filtration.
* **Matplotlib**

Matplotlib is a [plotting](https://en.wikipedia.org/wiki/Plotter) [library](https://en.wikipedia.org/wiki/Library_(computer_science)) for the [Python](https://en.wikipedia.org/wiki/Python_(programming_language)) programming language and its numerical mathematics extension [NumPy](https://en.wikipedia.org/wiki/NumPy). It provides an [object-oriented](https://en.wikipedia.org/wiki/Object-oriented_programming) [API](https://en.wikipedia.org/wiki/API) for embedding plots into applications using general-purpose [GUI toolkits](https://en.wikipedia.org/wiki/GUI_toolkit) like [Tkinter](https://en.wikipedia.org/wiki/Tkinter" \o "Tkinter), [wxPython](https://en.wikipedia.org/wiki/WxPython" \o "WxPython), [Qt](https://en.wikipedia.org/wiki/Qt_(software)), or [GTK+](https://en.wikipedia.org/wiki/GTK%2B). There is also a [procedural](https://en.wikipedia.org/wiki/Procedural_programming) "pylab" interface based on a [state machine](https://en.wikipedia.org/wiki/State_machine) (like [OpenGL](https://en.wikipedia.org/wiki/OpenGL)), designed to closely resemble that of [MATLAB](https://en.wikipedia.org/wiki/MATLAB), though its use is discouraged [SciPy](https://en.wikipedia.org/wiki/SciPy) makes use of Matplotlib.

Matplotlib was originally written by [John D. Hunter](https://en.wikipedia.org/wiki/John_D._Hunter), has an active development community, and is distributed under a [BSD-style license](https://en.wikipedia.org/wiki/BSD_licenses). Michael Droettboom was nominated as matplotlib's lead developer shortly before John Hunter's death in August 2012, and further joined by Thomas Caswell.

As of 23 June 2017, matplotlib 2.0.x supports Python versions 2.7 through 3.6. Matplotlib 1.2 is the first version of matplotlib to support Python 3.x. Matplotlib 1.4 is the last version of Matplotlib to support Python 2.6.

* **NumPy**

NumPy is a library for the [Python programming language](https://en.wikipedia.org/wiki/Python_(programming_language)), adding support for large, multi-dimensional [arrays](https://en.wikipedia.org/wiki/Array_data_structure) and [matrices](https://en.wikipedia.org/wiki/Matrix_(math)), along with a large collection of [high-level](https://en.wikipedia.org/wiki/High-level_programming_language) [mathematical](https://en.wikipedia.org/wiki/Mathematics) [functions](https://en.wikipedia.org/wiki/Function_(mathematics)) to operate on these arrays. The ancestor of NumPy, Numeric, was originally created by [Jim Hugunin](https://en.wikipedia.org/wiki/Jim_Hugunin) with contributions from several other developers. In 2005, [Travis Oliphant](https://en.wikipedia.org/wiki/Travis_Oliphant) created NumPy by incorporating features of the competing Numarray into Numeric, with extensive modifications. NumPy is [open-source software](https://en.wikipedia.org/wiki/Open-source_software) and has many contributors.

* **Scikit-learn**

Scikit-learn (formerly scikits.learn) is a [free software](https://en.wikipedia.org/wiki/Free_software) [machine learning](https://en.wikipedia.org/wiki/Machine_learning) [library](https://en.wikipedia.org/wiki/Library_(computing)) for the [Python](https://en.wikipedia.org/wiki/Python_(programming_language)) programming language. It features various [classification](https://en.wikipedia.org/wiki/Statistical_classification), [regression](https://en.wikipedia.org/wiki/Regression_analysis) and [clustering](https://en.wikipedia.org/wiki/Cluster_analysis) algorithms including [support vector machines](https://en.wikipedia.org/wiki/Support_vector_machine), [random forests](https://en.wikipedia.org/wiki/Random_forests), [gradient boosting](https://en.wikipedia.org/wiki/Gradient_boosting), [*k*-means](https://en.wikipedia.org/wiki/K-means_clustering) and [DBSCAN](https://en.wikipedia.org/wiki/DBSCAN), and is designed to interoperate with the Python numerical and scientific libraries [NumPy](https://en.wikipedia.org/wiki/NumPy) and [SciPy](https://en.wikipedia.org/wiki/SciPy). The scikit-learn project started as scikits.learn, a [Google Summer of Code](https://en.wikipedia.org/wiki/Google_Summer_of_Code) project by [David Cournapeau](https://en.wikipedia.org/wiki/David_Cournapeau).