DATA SCIENCE

(Prediction of heart disease occurrence)

Summer Internship Report Submitted in partial fulfillment

of the requirement for undergraduate degree of

Bachelor of Technology

In

Computer Science and Engineering

By

CHAMA SRI CHANDANA

221710313010

Under the Guidance of
Mr. _____
Assistant Professor

GITAM
(DEEMED TO BE UNIVERSITY)
(Estd. u/s 3 of the UGC Act, 1956)
VISAKHAPATNAM ✺ HYDERABAD ✺ BENGALURU

Department Of Computer Science And Engineering
GITAM School of Technology
GITAM (Deemed to be University)
Hyderabad-502329
June 2020

DECLARATION

I submit this industrial training work entitled “PREDICTION OF HEART DISEASE OCCURANCE” to GITAM (Deemed To Be University), Hyderabad in partial fulfillment of the requirements for the award of the degree of “Bachelor of Technology” in “Computer Science and Engineering”. I declare that it was carried out independently by me under the guidance of _____, _____, GITAM (Deemed To Be University), Hyderabad, India.

The results embodied in this report have not been submitted to any other University or Institute for the award of any degree or diploma.

Place:HYDERABAD

CHAMA SRI CHANDANA
221710313010

CERTIFICATE

CERTIFICATE

ACKNOWLEDGEMENT

Apart from my effort, the success of this internship largely depends on the encouragement and guidance of many others. I take this opportunity to express my gratitude to the people who have helped me in the successful competition of this internship.

I would like to thank Dr. N. Siva Prasad, Pro Vice Chancellor, GITAM Hyderabad and Dr. CH. Sanjay, Principal, GITAM Hyderabad

I am grateful to our project guide Mr. Dinesh Kumar Jooshetti at Promize IT Services Pvt Ltd. ,Hyderabad for the guidance, inspiration and constructive suggestions that helped us in preparation of this project.

I would like to thank respected Dr. S.Phani Kumar, Head of the Department of Computer Science department for giving me such a wonderful opportunity to expand my knowledge for my own branch and giving me guidelines to present an internship report. It helped me a lot to realize what we study for.

I wish to express our warm and grateful thanks to our project coordinator for the guidance and assistance he provided in completion of our project successfully.

I would also like to thank my friends who helped me to make my work more organized and well-stacked till the end.

CHAMA SRI CHANDANA

221710313010

ABSTRACT

Heart is the next major organ comparing to brain which has more priority in Human body. It pumps the blood and supplies to all organs of the whole body. Prediction of occurrences of heart diseases in medical field is significant work. Data analytics is useful for prediction from more information and it helps medical centre to predict of various disease. Huge amount of patient related data is maintained on monthly basis. The stored data can be useful for source of predicting the occurrence of future disease. Some of the data mining and machine learning techniques are used to predict the heart disease, such as Artificial Neural Network (ANN), Decision tree, Fuzzy Logic, K-Nearest Neighbour (KNN), Naïve Bayes and Support Vector, Machine (SVM), Logistic Regression. This project provides an insight of the some of the existing machine algorithms and it gives an overall summary of the existing work.

CONTENTS

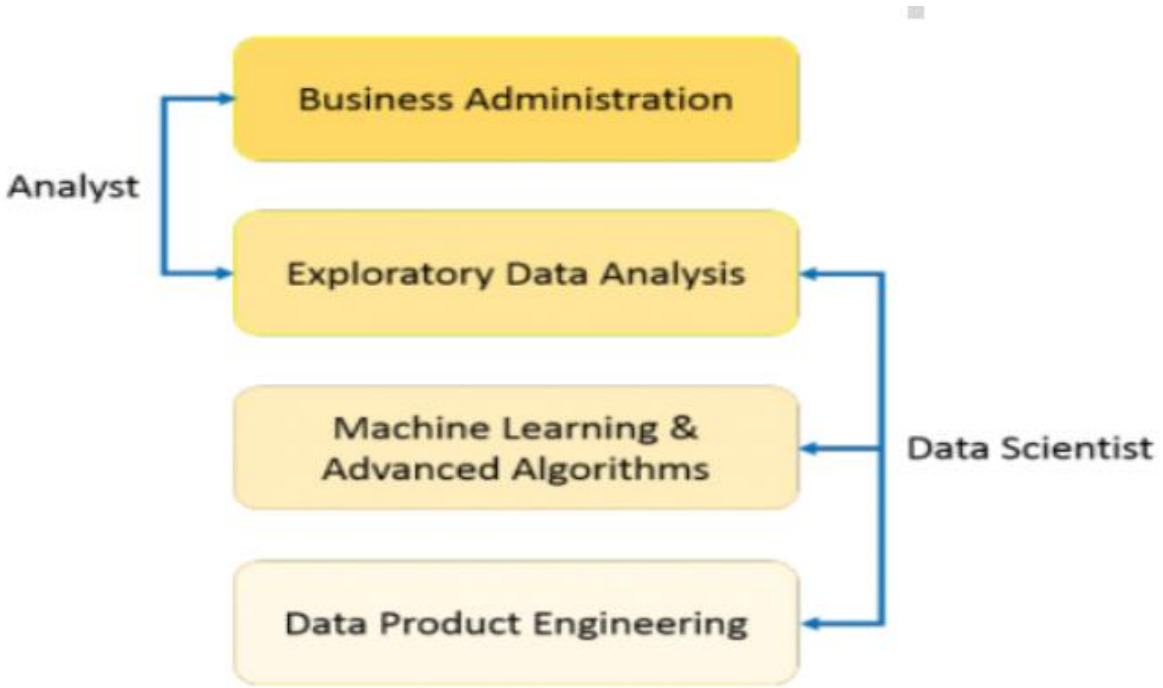
CHAPTER 1

INFORMATION ABOUT DATA SCIENCE

1.1.1 WHAT IS DATA SCIENCE?

Data Science is a blend of various tools, algorithms, and machine learning principles with the goal to discover hidden patterns from the raw data. How is this different from what statisticians have been doing for years?

The answer lies in the difference between explaining and predicting.



As you can see from the above image, a Data Analyst usually explains what is going on by processing history of the data. On the other hand, Data Scientist not only does the exploratory analysis to discover insights from it, but also uses various advanced machine learning algorithms to identify the occurrence of a particular event in the future. A Data Scientist will look at the data from many angles, sometimes angles not known earlier.

1.1.2 NEED OF DATA SCIENCE

Data Science is primarily used to make decisions and predictions making use of predictive causal analytics, prescriptive analytics (predictive plus decision science) and machine learning.

- Predictive causal analytics – If you want a model which can predict the possibilities of a particular event in the future, you need to apply predictive causal analytics. Say, if you are providing money on credit, then the probability of customers making future credit payments on time is a matter of concern for you. Here, you can build a model which can perform predictive analytics on the payment history of the customer to predict if the future payments will be on time or not.

- Prescriptive analytics: If you want a model which has the intelligence of taking its own decisions and the ability to modify it with dynamic parameters, you certainly need prescriptive analytics for it. This relatively new field is all about providing advice. In other terms, it not only predicts but suggests a range of prescribed actions and associated outcomes.
  The best example for this is Google's self-driving car which I had discussed earlier too. The data gathered by vehicles can be used to train self-driving cars. You can run algorithms on this data to bring intelligence to it. This will enable your car to take decisions like when to turn, which path to take, when to slow down or speed up.

- Machine learning for making predictions — If you have transactional data of a finance company and need to build a model to determine the future trend, then machine learning algorithms are the best bet. This falls under the paradigm of supervised learning. It is called supervised because you already have the data based on which you can train your

- machines. For example, a fraud detection model can be trained using a historical record of fraudulent purchases.

- Machine learning for pattern discovery — If you don't have the parameters based on which you can make predictions, then you need to find out the hidden patterns within the dataset to be able to make meaningful predictions. This is nothing but the

unsupervised model as you don’t have any predefined labels for grouping. The most common algorithm used for pattern discovery is Clustering.

Let’s say you are working in a telephone company and you need to establish a network by putting towers in a region. Then, you can use the clustering technique to find those tower locations which will ensure that all the users receive optimum signal strength.

Let’s see how the proportion of above-described approaches differ for Data Analysis as well as Data Science. As you can see in the image below, Data Analysis includes descriptive analytics and prediction to a certain extent. On the other hand, Data Science is more about Predictive Causal Analytics and Machine Learning.

1.1.3 USES OF DATA SCIENCE

1. Medical Image Analysis

Procedures such as detecting tumors, artery stenosis, organ delineation employ various different methods and frameworks like MapReduce to find optimal parameters for tasks like lung texture classification. It applies machine learning methods, support vector machines (SVM), content-based medical image indexing, and wavelet analysis for solid texture classification.

2. Genetics & Genomics

Data Science applications also enable an advanced level of treatment personalization through research in genetics and genomics. The goal is to understand the impact of the DNA on our health and find individual biological connections between genetics, diseases, and drug response. Data science techniques allow integration of different kinds of data with genomic data in the disease research, which provides a deeper understanding of genetic issues in reactions to particular drugs and diseases. As soon as we acquire reliable personal genome data, we will achieve a deeper understanding of the human DNA. The advanced genetic risk prediction will be a major step towards more individual care.

3. Drug Development

The drug discovery process is highly complicated and involves many disciplines. The greatest ideas are often bounded by billions of testing, huge financial and time expenditure. On average, it takes twelve years to make an official submission.

Data science applications and machine learning algorithms simplify and shorten this process, adding a perspective to each step from the initial screening of drug compounds to the prediction of the success rate based on the biological factors. Such algorithms can forecast how the compound will act in the body using advanced mathematical modeling and simulations instead of the "lab experiments". The idea behind the computational drug discovery is to create computer model simulations as a biologically relevant network simplifying the prediction of future outcomes with high accuracy.

4. **Virtual assistance for patients and customer support**

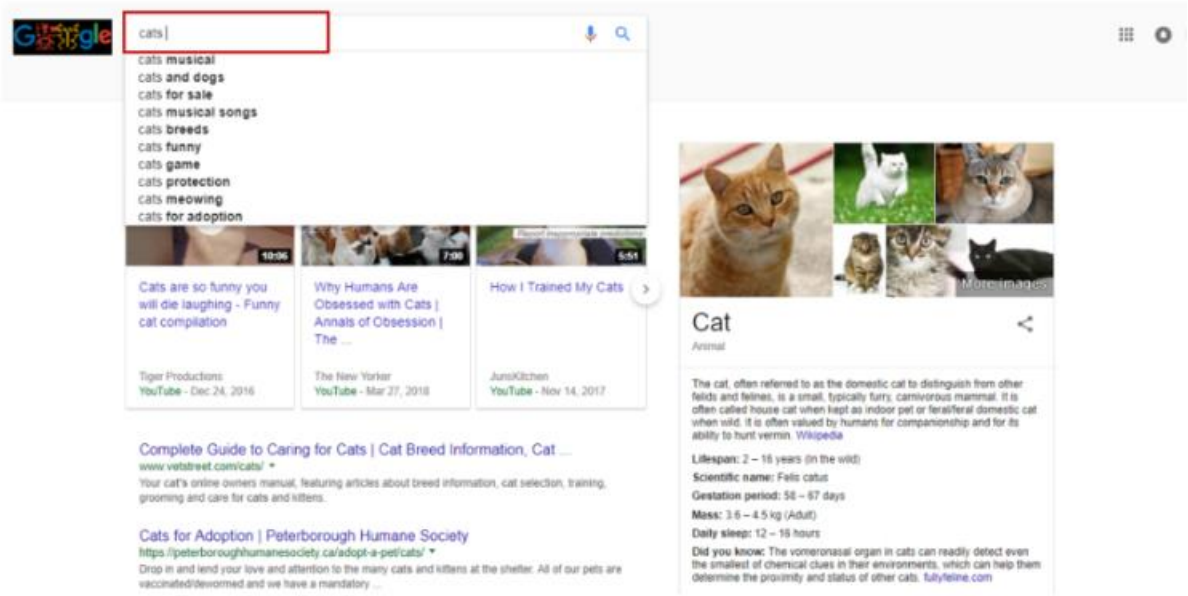
Optimization of the clinical process builds upon the concept that for many cases it is not actually necessary for patients to visit doctors in person. A mobile application can give a more effective solution by *bringing the doctor to the patient instead*.The AI-powered mobile apps can provide basic healthcare support, usually as chatbots. You simply describe your symptoms, or ask questions, and then receive key information about your medical condition derived from a wide network linking symptoms to causes. Apps can remind you to take your medicine on time, and if necessary, assign an appointment with a doctor.This approach promotes a healthy lifestyle by

encouraging patients to make healthy decisions, saves their time waiting in line for an appointment, and allows doctors to focus on more critical cases.

5.Internet Search

Now, this is probably the first thing that strikes your mind when you think Data Science Applications.When we speak of search, we think 'Google'. Right? But there are many other

search engines like Yahoo, Bing, Ask, AOL, and so on. All these search engines (including Google) make use of data science algorithms to deliver the best result for our searched query in a fraction of seconds. Considering the fact that, Google processes more than 20 petabytes of data every day.

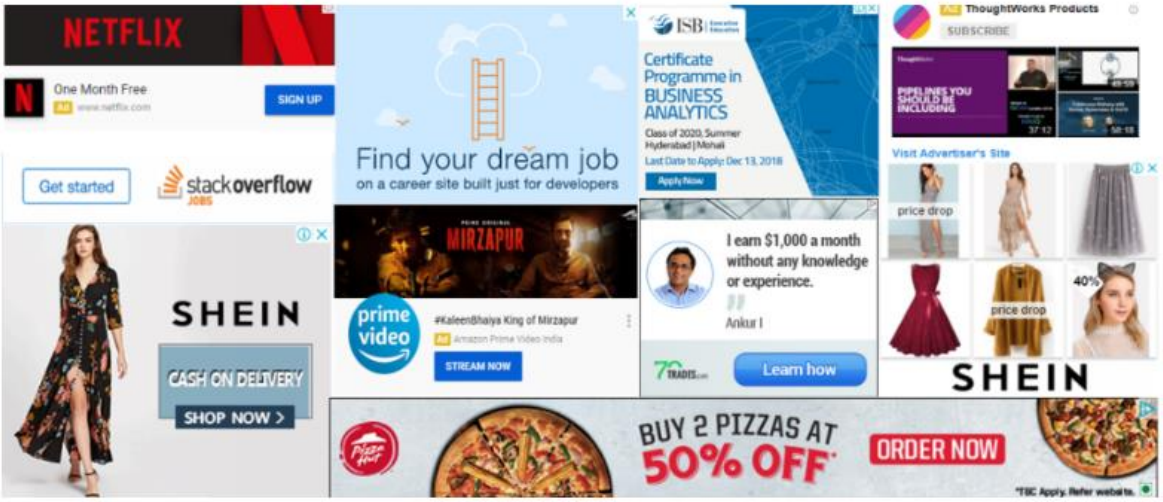


6.Targeted Advertising

If you thought Search would have been the biggest of all data science applications, here is a challenger – the entire digital marketing spectrum. Starting from the display banners on various websites to the digital billboards at the airports – almost all of them are decided by using data science algorithms.

This is the reason why digital ads have been able to get a lot higher CTR (Call-Through Rate) than traditional advertisements. They can be targeted based on a user's past behavior.

This is the reason why you might see ads of Data Science Training Programs while I see an ad of apparels in the same place at the same time.


NETFLIX
One Month Free
SIGN UP
Get started
stackoverflow
JOBS
SHEIN
CASH ON DELIVERY
SHOP NOW >
Find your dream job
on a career site built just for developers
ISB
Certificate Programme in
BUSINESS ANALYTICS
Class of 2020, Summer
Hyderabad | Mohali
Last Date to Apply: Dec 13, 2018
Apply Now
MIRZAPUR
prime video
#KaleenBhaiya King of Mirzapur
STREAM NOW
I earn $1,000 a month without any knowledge or experience.
Ankur I
Learn how
ThoughtWorks Products
SUBSCRIBE
PIPELINES YOU SHOULD BE INCLUDING
Visit Advertiser's Site
price drop
price drop
40%
SHEIN
BUY 2 PIZZAS AT
50% OFF
ORDER NOW
*T&C Apply. Refer website.

CHAPTER 2

INFORMATION ABOUT MACHINE LEARNING

2.1 INTRODUCTION:

Machine Learning (ML) is the scientific study of algorithms and statistical models that computer systems use in order to perform a specific task effectively without using explicit instructions, relying on patterns and inference instead. It is seen as a subset of Artificial Intelligence (AI).

2.2 IMPORTANCE OF MACHINE LEARNING:

Consider some of the instances where machine learning is applied: the self-driving Google car, cyber fraud detection, online recommendation engines—like friend suggestions on Facebook, Netflix showcasing the movies and shows you might like, and "more items to consider" and "get yourself a little something" on Amazon—are all examples of applied machine learning. All these examples echo the vital role machine learning has begun to take in today's data-rich world.

Machines can aid in filtering useful pieces of information that help in major advancements, and we are already seeing how this technology is being implemented in a wide variety of industries.

With the constant evolution of the field, there has been a subsequent rise in the uses, demands, and importance of machine learning. Big data has become quite a buzzword in the last few years; that's in part due to increased sophistication of machine learning, which helps analyze those big chunks of big data. Machine learning has also changed the way data extraction, and interpretation is done by involving automatic sets of generic methods that have replaced traditional statistical techniques.

The process flow depicted here represents how machine learning works

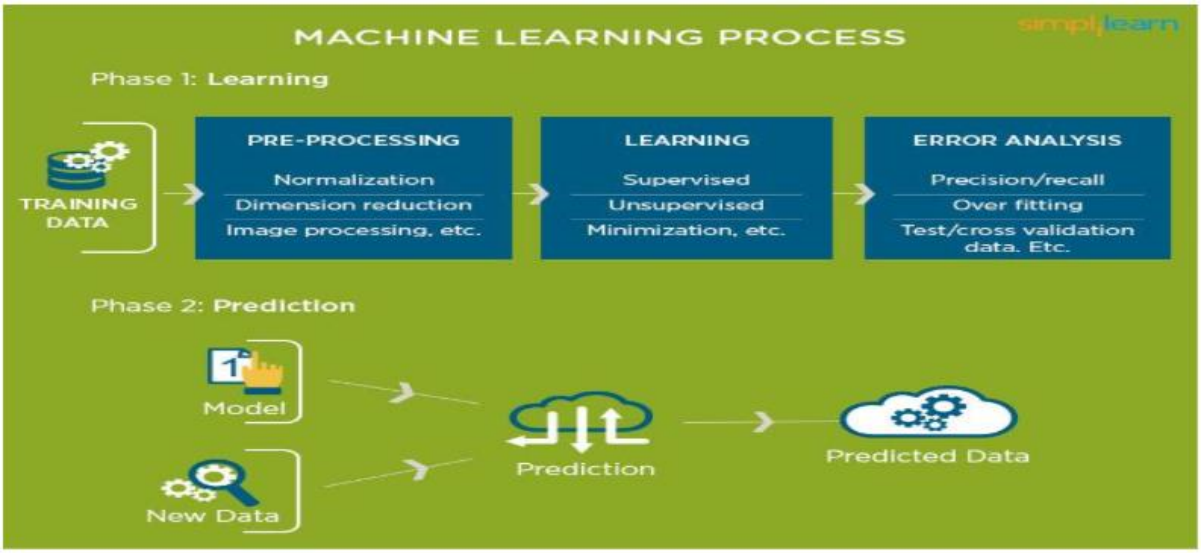


Figure 1 : The Process Flow

2.3 USES OF MACHINE LEARNING:

Earlier in this article, we mentioned some applications of machine learning. To understand the concept of machine learning better, let's consider some more examples: web search results, real-time ads on web pages and mobile devices, email spam filtering, network intrusion detection, and pattern and image recognition. All these are by-products of applying machine learning to analyze huge volumes of data Traditionally, data analysis was always being characterized by trial and error, an approach that becomes impossible when data sets are large and heterogeneous. Machine learning comes as the solution to all this chaos by proposing clever alternatives to analyzing huge volumes of data.

By developing fast and efficient algorithms and data-driven models for real-time processing of data, machine learning can produce accurate results and analysis.

2.4 TYPES OF LEARNING ALGORITHMS:

The types of machine learning algorithms differ in their approach, the type of data they input and output, and the type of task or problem that they are intended to solve.

Supervised Learning :

When an algorithm learns from example data and associated target responses that can consist of numeric values or string labels, such as classes or tags, in order to later predict the correct response when posed with new examples comes under the category of supervised learning. Supervised machine learning algorithms uncover insights, patterns, and relationships from a

labelled training dataset – that is, a dataset that already contains a known value for the target variable for each record. Because you provide the machine learning algorithm with the correct answers for a problem during training, it is able to "learn" how the rest of the features relate to the target, enabling you to uncover insights and make predictions about future outcomes based on historical data.

Examples of Supervised Machine Learning Techniques are Regression, in which the algorithm returns a numerical target for each example, such as how much revenue will be generated from a new marketing campaign.

Classification, in which the algorithm attempts to label each example by choosing between two or more different classes. Choosing between two classes is called binary classification, such as determining whether or not someone will default on a loan. Choosing between more than two classes is referred to as multiclass classification.

Unsupervised Learning:

When an algorithm learns from plain examples without any associated response, leaving to the algorithm to determine the data patterns on its own. This type of algorithm tends to restructure the data into something else, such as new features that may represent a class or a new series of uncorrelated values. They are quite useful in providing humans with insights into the meaning of data and new useful inputs to supervised machine learning algorithms.

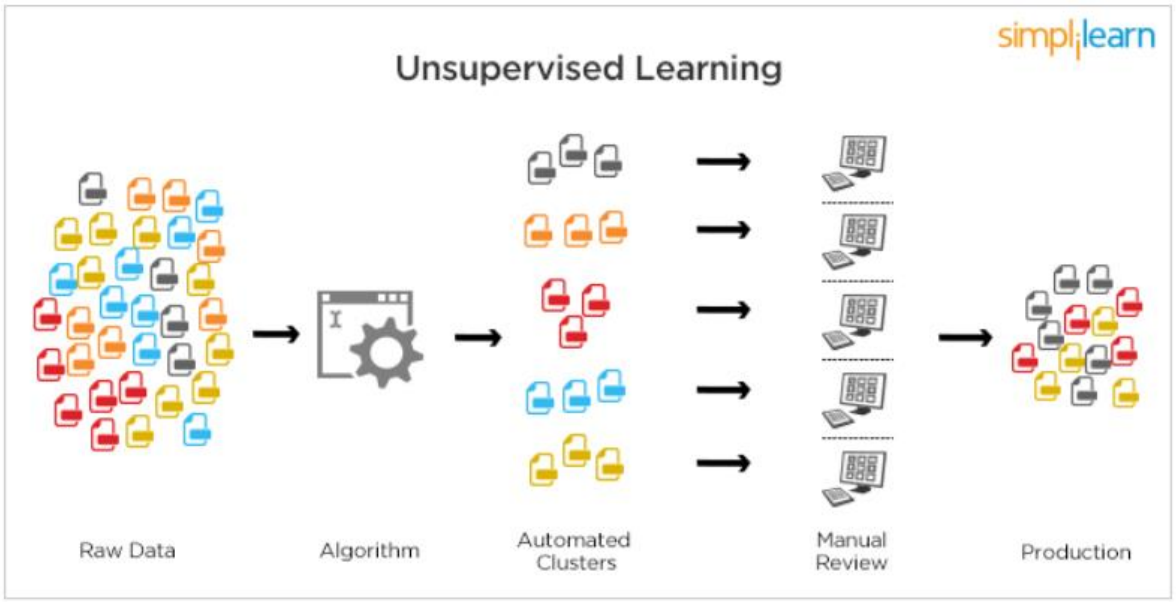


Figure 2 : Unsupervised Learning

Popular techniques where unsupervised learning is used also include self-organizing maps, nearest neighbor mapping, singular value decomposition, and k-means clustering. Basically, online recommendations, identification of data outliers, and segment text topics are all examples of unsupervised learning.

Semi Supervised Learning:

As the name suggests, semi-supervised learning is a bit of both supervised and unsupervised learning and uses both labeled and unlabeled data for training. In a typical scenario, the algorithm would use a small amount of labeled data with a large amount of unlabeled data.

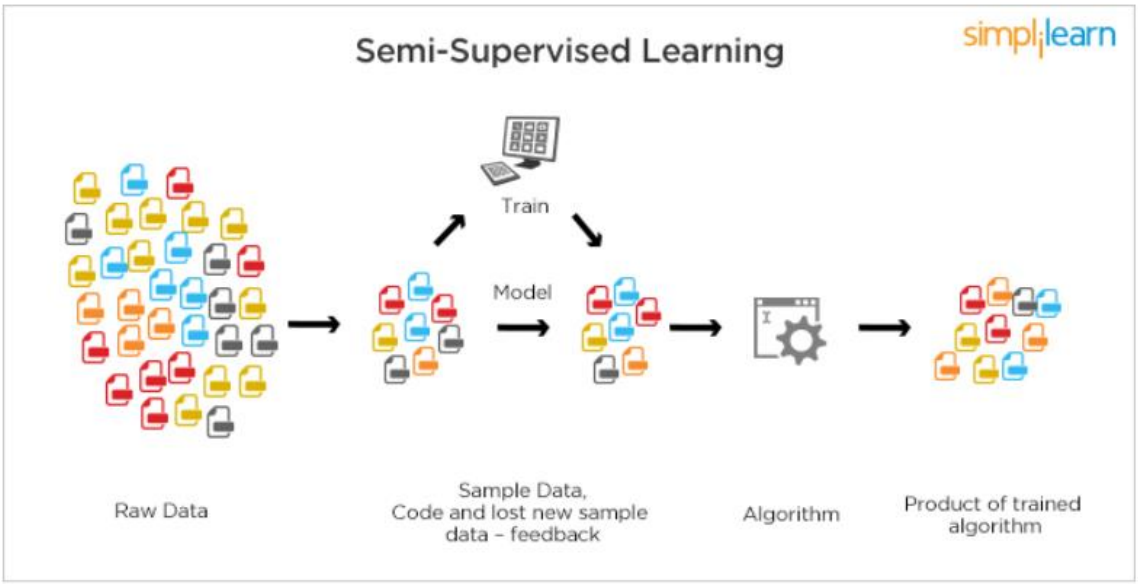


Figure 3 : Semi Supervised Learning

2.5 RELATION BETWEEN DATA MINING, MACHINE LEARNING AND DEEP LEARNING:

Machine learning and data mining use the same algorithms and techniques as data mining, except the kinds of predictions vary. While data mining discovers previously unknown patterns and knowledge, machine learning reproduces known patterns and knowledge—and further automatically applies that information to data, decision-making, and actions.

Deep learning, on the other hand, uses advanced computing power and special types of neural networks and applies them to large amounts of data to learn, understand, and identify complicated patterns. Automatic language translation and medical diagnoses are examples of deep learning.

CHAPTER 3
INFORMATION ABOUT PYTHON

Basic programming language used for machine learning is : PYTHON

3.1 INTRODUCTION TO PYTHON:

● Python is a high-level, interpreted, interactive and object-oriented scripting language.

● Python is a general purpose programming language that is often applied in scripting roles

● Python is Interpreted: Python is processed at runtime by the interpreter. You do not need to compile your program before executing it. This is like PERL and PHP.

● Python is Interactive: You can sit at a Python prompt and interact with the interpreter directly to write your programs.

● Python is Object-Oriented: Python supports the Object-Oriented style or technique of programming that encapsulates code within objects.

HISTORY OF PYTHON:

● Python was developed by GUIDO VAN ROSSUM in early 1990's

● Its latest version is 3.7 , it is generally called as python

3.2 FEATURES OF PYTHON:

● Easy-to-learn: Python has few keywords, simple structure, and a clearly defined syntax, This allows the student to pick up the language quickly.

● Easy-to-read: Python code is more clearly defined and visible to the eyes.

● Easy-to-maintain: Python's source code is fairly easy-to-maintaining.

● A broad standard library: Python's bulk of the library is very portable and cross-platform compatible on UNIX, Windows, and Macintosh.

● Portable: Python can run on a wide variety of hardware platforms and has the same interface on all platforms.

● Extendable: You can add low-level modules to the Python interpreter. These modules enable programmers to add to or customize their tools to be more efficient.

● Databases: Python provides interfaces to all major commercial databases.

● GUI Programming: Python supports GUI applications that can be created and ported to many system calls, libraries and windows systems, such as Windows MFC, Macintosh, and the X Window system of Unix.

3.3 HOW TO SETUP PYTHON:

● Python is available on a wide variety of platforms including Linux and Mac OS X. Let's understand how to set up our Python environment.

● The most up-to-date and current source code, binaries, documentation, news, etc., is available on the official website of Python.

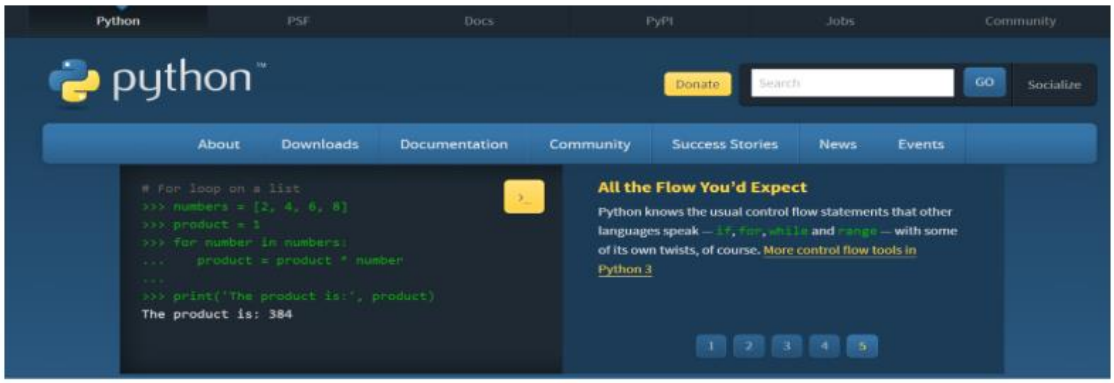
Installation(using python IDLE):

● Installing python is generally easy, and nowadays many Linux and Mac OS distributions include a recent python.

● Download python from www.python.org

● When the download is completed, double click the file and follow the instructions to install it.

● When python is installed, a program called IDLE is also installed along with it. It provides a graphical user interface to work with python.



Installation(using Anaconda):

● Python programs are also executed using Anaconda.

● Anaconda is a free open source distribution of python for large scale data processing, predictive analytics and scientific computing.

● Conda is a package manager quickly installs and manages packages.

● In WINDOWS:

● In windows

- ● Step 1: Open Anaconda.com/downloads in web browser.
- ● Step 2: Download python 3.4 version for (32-bitgraphic installer/64 -bit graphic installer)
- ● Step 3: select installation type(all users)
- ● Step 4: Select path(i.e. add anaconda to path & register anaconda as default python 3.4) next click install and next click finish
- ● Step 5: Open jupyter notebook (it opens in default browser).

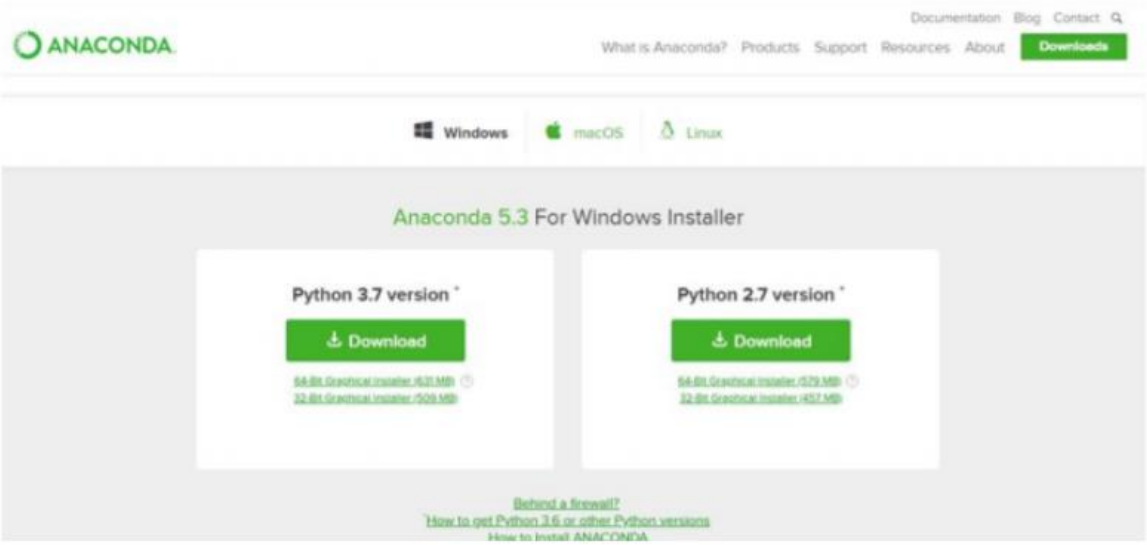


Figure 5 : Anaconda download

Figure 6 : Jupyter notebook

3.4 PYTHON VARIABLE TYPES:

● Variables are nothing but reserved memory locations to store values. This means that when you create a variable you reserve some space in memory.

● Variables are nothing but reserved memory locations to store values.

● Based on the data type of a variable, the interpreter allocates memory and decides what can be stored in the reserved memory.

● Python variables do not need explicit declaration to reserve memory space. The declaration happens automatically when you assign a value to a variable.

● Python has various standard data types that are used to define the operations possible on them and the storage method for each of them.

● Python has five standard data types –

- o Numbers
- o Strings
- o Lists
- o Tuples
- o Dictionary

Python Numbers:

● Number data types store numeric values. Number objects are created when you assign a value to them.

● Python supports four different numerical types − int (signed integers) long (long integers, they can also be represented in octal and hexadecimal) float (floating point real values) complex (complex numbers).

Python Strings:

● Strings in Python are identified as a contiguous set of characters represented in the quotation marks.

● Python allows for either pairs of single or double quotes.

● Subsets of strings can be taken using the slice operator ([ ] and [:]) with indexes starting at 0 in the beginning of the string and working their way from -1 at the end.

● The plus (+) sign is the string concatenation operator and the asterisk (*) is the repetition operator.

Python Lists:

● Lists are the most versatile of Python's compound data types

● A list contains items separated by commas and enclosed within square brackets.

● To some extent, lists are similar to arrays in C. One difference between them is that all the items belonging to a list can be of different data type.

● The values stored in a list can be accessed using the slice operator ([ ] and [:]) with indexes starting at 0 in the beginning of the list and working their way to end -1.

● The plus (+) sign is the list concatenation operator, and the asterisk (*) is the repetition operator.

Python Tuples:

● A tuple is another sequence data type that is similar to the list.

● A tuple consists of a number of values separated by commas. Unlike lists, however, tuples are enclosed within parentheses.

● The main differences between lists and tuples are: Lists are enclosed in brackets ([ ]) and their elements and size can be changed, while tuples are enclosed in parentheses (( )) and cannot be updated.

● Tuples can be thought of as read-only lists.

● For example − Tuples are fixed size in nature whereas lists are dynamic. In other words, a tuple is immutable whereas a list is mutable. You can't add elements to a tuple. Tuples have no append or extend method. You can't remove elements from a tuple. Tuples have no remove or pop method.

Python Dictionary:

● Python's dictionaries are kind of hash table type. They work like associative arrays or hashes found in Perl and consist of key-value pairs. A dictionary key can be almost any Python type, but are usually numbers or strings. Values, on the other hand, can be any arbitrary Python object.

● Dictionaries are enclosed by curly braces ({ }) and values can be assigned and accessed using square braces ([]).

● You can use numbers to "index" into a list, meaning you can use numbers to find out what's in lists. You should know this about lists by now, but make sure you understand that you can only use numbers to get items out of a list.

● What a dict does is let you use anything, not just numbers. Yes, a dict associates one thing to another, no matter what it is.

3.5 PYTHON FUNCTION:

Defining a Function:

You can define functions to provide the required functionality. Here are simple rules to define a function in Python. Function blocks begin with the keyword def followed by the function name and parentheses (i.e.())

Any input parameters or arguments should be placed within these parentheses. You can also define parameters inside these parentheses.

The code block within every function starts with a colon (:) and is indented. The statement returns [expression] exits a function, optionally passing back an expression to the caller. A return statement with no arguments is the same as return None.

Calling a Function:

Defining a function only gives it a name, specifies the parameters that are to be included in the function and structures the blocks of code. Once the basic structure of a function is finalized, you can execute it by calling it from another function or directly from the Python prompt.

3.6 PYTHON USING OOP's CONCEPTS:

Class:

● Class: A user-defined prototype for an object that defines a set of attributes that characterize any object of the class. The attributes are data members (class variables and instance variables) and methods, accessed via dot notation.

● Class variable: A variable that is shared by all instances of a class. Class variables are defined within a class but outside any of the class's methods. Class variables are not used as frequently as instance variables are.

● Data member: A class variable or instance variable that holds data associated with a class and its objects.

● Instance variable: A variable that is defined inside a method and belongs only to the current instance of a class.

● Defining a Class:

o We define a class in a very similar way how we define a function.

o Just like a function ,we use parentheses and a colon after the class name(i.e. ():) when we define a class. Similarly, the body of our class is indented like a functions body is.

```
def my_function():
    # the details of the
    # function go here
```

```
class MyClass():
    # the details of the
    # class go here
```

Figure 7 : Defining a Class

__init__ method in Class:

● The init method — also called a constructor — is a special method that runs when an instance is created so we can perform any tasks to set up the instance.

● The init method has a special name that starts and ends with two underscores:__init__().

CHAPTER 4

PREDICTION OF HEART DISEASE OCCURRENCE)

(INFORMATION ABOUT THE PROJECT)

1.1Project Requirements

1. Python IDE
2. Data set

4.1 Packages Used

1. numpy: To work with arrays

2.pandas: To work with csv files and dataframes

3.matplotlib: To create charts using pyplot

4. seaborn:

```
[3]  import numpy as np # linear algebra
     import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)
     import matplotlib.pyplot as plt
     import seaborn as sns
```

For processing the data, we need to import a few libraries. To split the available dataset for testing and training, train_test_split method is used.

4.2 PROBLEM STATEMENT

To predict whether a person is suffering from Heart Disease or not, taking into consideration the features of the existing dataset.

4.3 DATASET DESCRIPTION

1. Age: displays the age of the individual.
2. Sex: displays the gender of the individual using the following format :

   1 = male

   0 = female

3. Chest-pain type: displays the type of chest-pain experienced by the individual using the following format :

    1 = typical angina

    2 = atypical angina

    3 = non — anginal pain

    4 = asymptotic

4. Resting Blood Pressure: displays the resting blood pressure value of an individual in mmHg (unit)
5. Serum Cholestrol: displays the serum cholesterol in mg/dl (unit)
6. Fasting Blood Sugar: compares the fasting blood sugar value of an individual with 120mg/dl.

    If fasting blood sugar > 120mg/dl then : 1 (true)

7. Resting ECG : displays resting electrocardiographic results

    0 = normal

    1 = having ST-T wave abnormality

    2= left ventricular hyperthrophy

8.Max heart rate achieved : displays the max heart rate achieved by an individual.

9.Exercise induced angina :

    1 = yes

    0 = no

10.ST depression induced by exercise relative to rest: displays the value which is an integer or float.

11.Peak exercise ST segment :

    1 = upsloping

    2 = flat

    2= downsloping

12.Number of major vessels (0–3) colored by flourosopy : displays the value as integer or float.

13. Thal : displays the thalassemia :

    3 = normal

    6 = fixed defect

    7 = reversible defect

14. Diagnosis of heart disease : Displays whether the individual is suffering from heart disease or not :

0 = absence

1, 2, 3, 4 = present.

4.4 OBJECTIVE OF CASE STUDY

To get a better understanding to make predictions on whether a person is suffering from Heart Disease or not. by considering the features of the data and using various machine language algorithm and suggest best suited machine language algorithm and provide the client with desired results

CHAPTER 5
DATA ANALYSIS

5.1 READING THE DATASET

Pandas in python provide an interesting method read_csv(). The read_csv function reads the entire dataset from a comma separated values file and we can assign it to a DataFrame to which all the operations can be performed. It helps us to access each and every row as well as columns and each and every value can be access using the dataframe.

data=pd.read_csv("/content/drive/My Drive/summer internship/heartdataset.csv",encoding="unicode_escape")

The dataset is loaded into the variable dataset. I'll just take a glimpse of the data using the describe() and info() methods before I actually start processing and visualizing it.
Looks like the dataset has a total of 303 rows and there are no missing values. There are a total of 14 features along with one target value which we wish to find.

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

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
count	302.00000	302.000000	302.000000	302.000000	302.000000	302.000000	302.000000	302.000000	302.000000	302.000000	302.000000	302.000000	302.000000	302.000000
mean	54.42053	0.682119	0.963576	131.602649	246.500000	0.149007	0.526490	149.569536	0.327815	1.043046	1.397351	0.718543	2.314570	0.543046
std	9.04797	0.466426	1.032044	17.563394	51.753489	0.356686	0.526027	22.903527	0.470196	1.161452	0.616274	1.006748	0.613026	0.498970
min	29.00000	0.000000	0.000000	94.000000	126.000000	0.000000	0.000000	71.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
25%	48.00000	0.000000	0.000000	120.000000	211.000000	0.000000	0.000000	133.250000	0.000000	0.000000	1.000000	0.000000	2.000000	0.000000
50%	55.50000	1.000000	1.000000	130.000000	240.500000	0.000000	1.000000	152.500000	0.000000	0.800000	1.000000	0.000000	2.000000	1.000000
75%	61.00000	1.000000	2.000000	140.000000	274.750000	0.000000	1.000000	166.000000	1.000000	1.600000	2.000000	1.000000	3.000000	1.000000
max	77.00000	1.000000	3.000000	200.000000	564.000000	1.000000	2.000000	202.000000	1.000000	6.200000	2.000000	4.000000	3.000000	1.000000

The scale of each feature column is different and quite varied as well. While the maximum for age reaches 77, the maximum of chol (serum cholesterol) is 564.So we need to normalize data before applying ML algorithms to get a uniform scale all over the data.

```
[41] data
```

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
0	52	1	0	125	212	0	1	168	0	1.0	2	2	3	0
1	53	1	0	140	203	1	0	155	1	3.1	0	0	3	0
2	70	1	0	145	174	0	1	125	1	2.6	0	0	3	0
3	61	1	0	148	203	0	1	161	0	0.0	2	1	3	0
4	62	0	0	138	294	1	1	106	0	1.9	1	3	2	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
723	68	0	2	120	211	0	0	115	0	1.5	1	0	2	1
733	44	0	2	108	141	0	1	175	0	0.6	1	0	2	1
739	52	1	0	128	255	0	1	161	1	0.0	2	1	3	0
843	59	1	3	160	273	0	0	125	0	0.0	2	0	2	0
878	54	1	0	120	188	0	1	113	0	1.4	1	1	3	0

302 rows × 14 columns

```
[ ]  data.shape
```

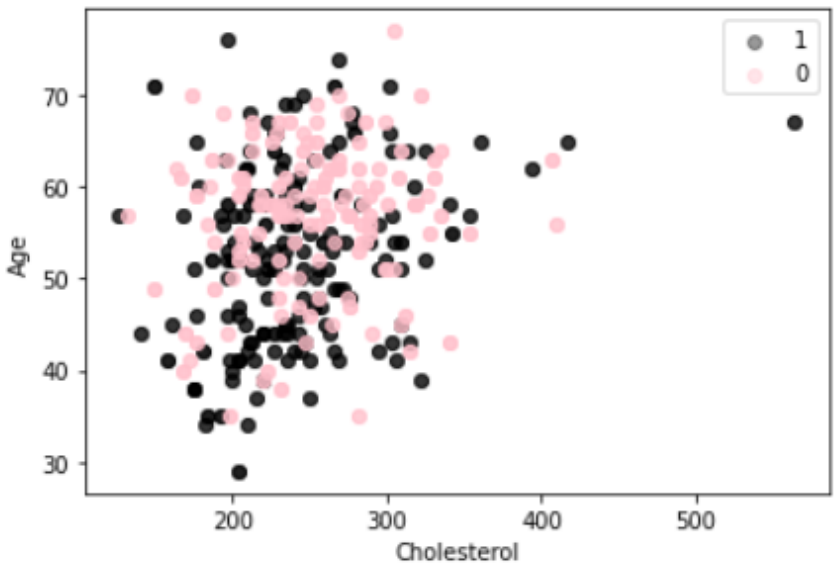
```
(1025, 14)
```

5.2Describing visualization of scatter plot :

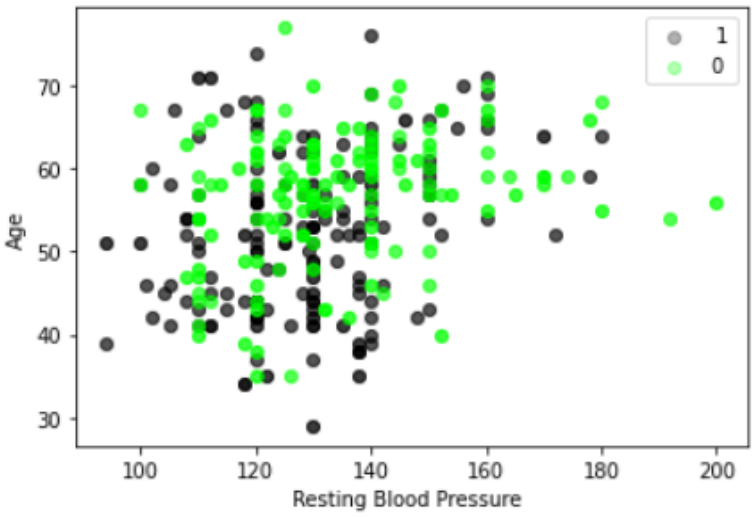
```
[ ]  #Split Data as M&B
     A = data[data.target == 1]
     B = data[data.target == 0]
```

The "goal" field refers to the presence of heart disease in the patient. Target = 1 => presence of heart disease Target = 0 => no of heart disease

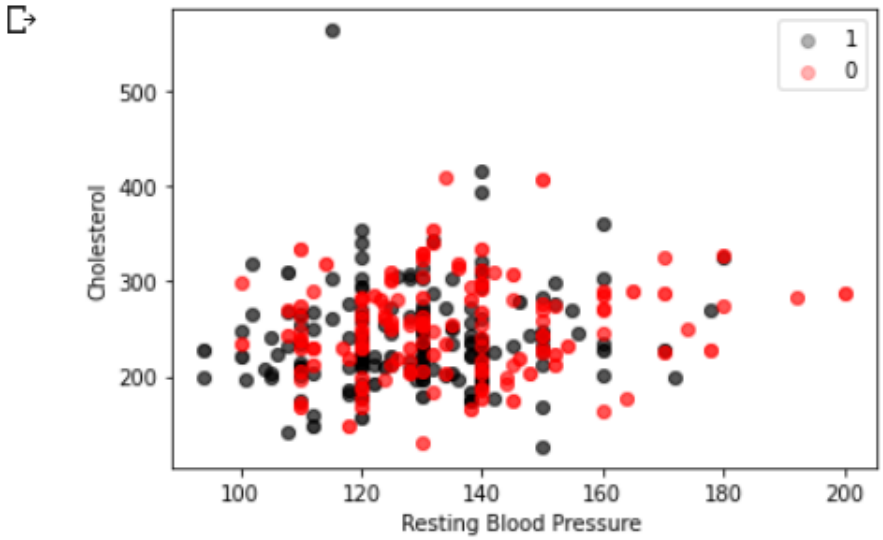
```
[ ]  #Visualization, Scatter Plot

     plt.scatter(A.chol,A.age,color = "black",label="1",alpha=0.4)
     plt.scatter(B.chol,B.age,color = "Pink",label="0",alpha=0.4)
     plt.xlabel("Cholesterol")
     plt.ylabel("Age")
     plt.legend()
     plt.show()
```




```
[ ]  #Visualization, Scatter Plot

     plt.scatter(A.trestbps,A.age,color = "Black",label=" 1",alpha=0.3)
     plt.scatter(B.trestbps,B.age,color = "Lime",label="0",alpha=0.3)
     plt.xlabel("Resting Blood Pressure ")
     plt.ylabel("Age")
     plt.legend()
     plt.show()
```



```
[ ]  #Visualization, Scatter Plot

     plt.scatter(A.trestbps,A.chol,color = "Black",label="1",alpha=0.3)
     plt.scatter(B.trestbps,B.chol,color = "red",label="0",alpha=0.3)
     plt.xlabel("Resting Blood Pressure ")
     plt.ylabel("Cholesterol")
     plt.legend()
     plt.show()
```



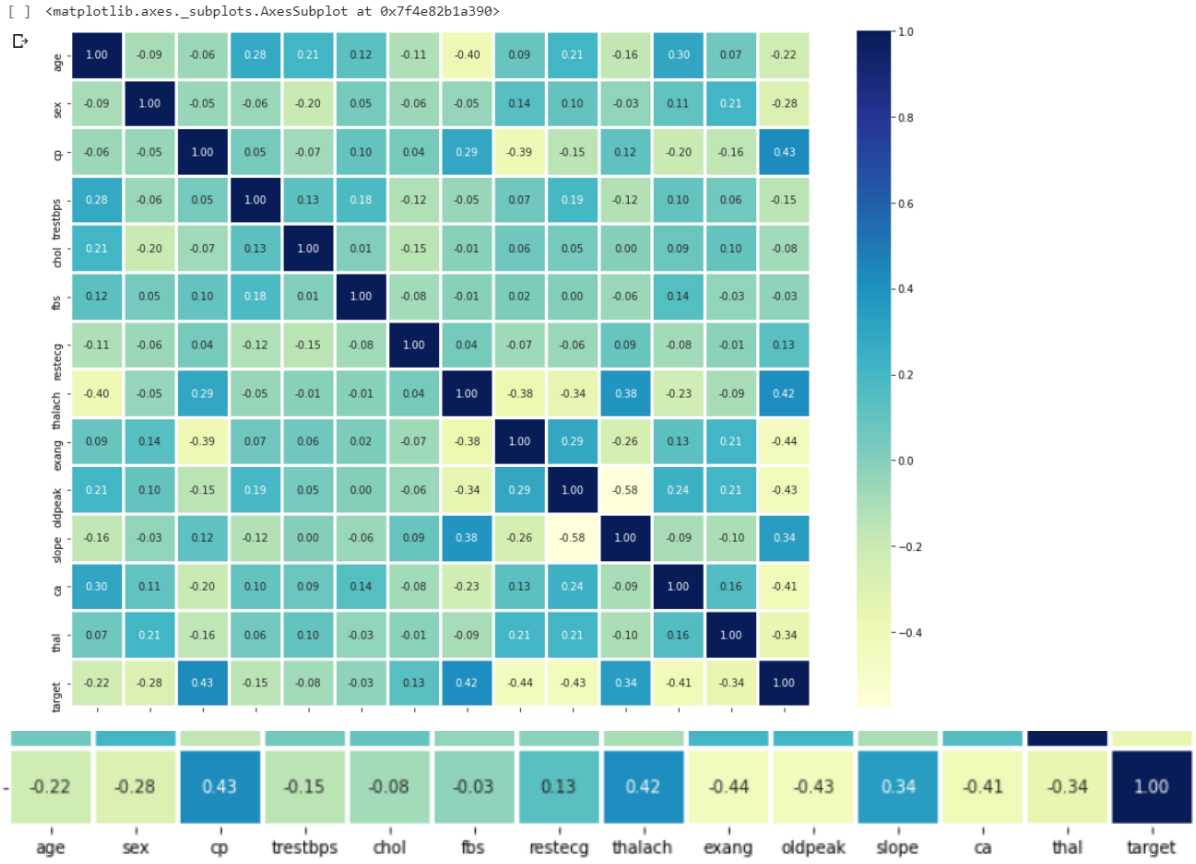
From the graph age and cholesterol the people who are having between 250-200 are of age between 45-60 are at highest risk of getting heart disease

From the graph resting blood pressure vs age it can be observe that both the chance of getting heart disease and not getting heart disease are nearly equal for the people who are having resting blood pressure between 120-140 irrespective of age.

From the scatter plot resting blood pressure vs cholesterol it can be observed that it can be observed that people having 250-300 and resting blood pressure more than 105 are at high chance of heart disease

5.3 Correlation Matrix

we can use visualizations to better understand our data and then look at any processing we might want to do**.**

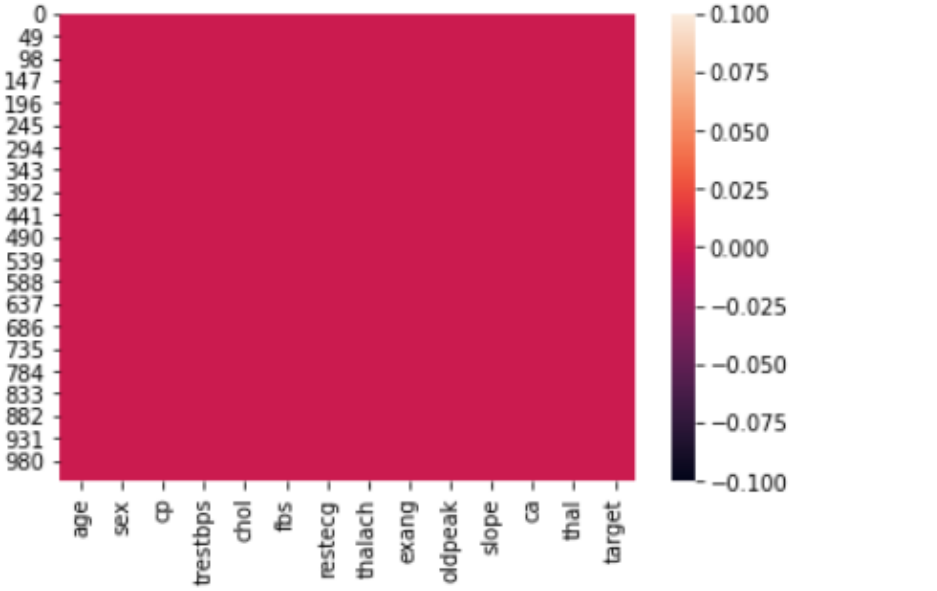


Taking a look at the correlation matrix above, it's easy to see that a few features have negative correlation with the target value while some have positive. Next, I'll take a look at the histograms for each variable.

5.4 Heatmap using seaborn

```
[ ]  sns.heatmap(data.isna())
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fbe223105c0>
```

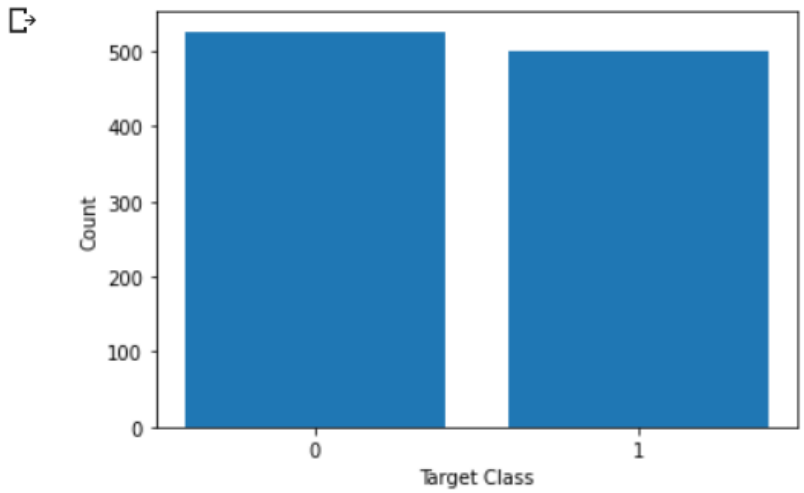


5.5 HISTOGRAM

Taking a look at the histograms above, I can see that each feature has a different range of distribution. Thus, using scaling before our predictions should be of great use. Also, the categorical features do stand out.

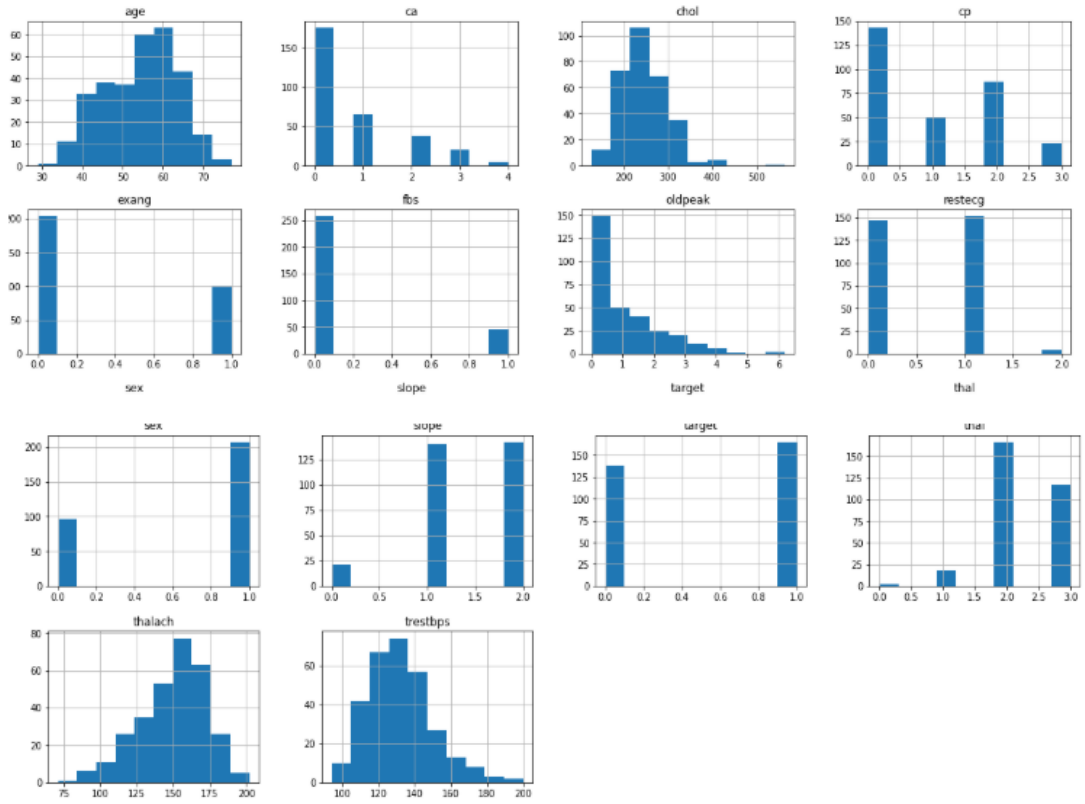
It's always a good practice to work with a dataset where the target classes are of approximately equal size. Thus, let's check for the same.

```
[ ]  plt.bar(data.target.unique(),data.target.value_counts())
     plt.xlabel('Target Class')
     plt.xticks([0,1])
     plt.ylabel('Count')
     plt.show()
```



```
heart.hist()
```

```
array([[<matplotlib.axes._subplots.AxesSubplot object at 0x7fa2a8251dd8>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7fa2a2b4da20>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7fa2a2affc88>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7fa2a2b34ef0>],
       [<matplotlib.axes._subplots.AxesSubplot object at 0x7fa2a2af4198>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7fa2a2aa7400>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7fa2a2a5c668>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7fa2a2a10898>],
       [<matplotlib.axes._subplots.AxesSubplot object at 0x7fa2a2a10908>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7fa2a2978da0>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7fa2a29acfd0>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7fa2a296e2b0>],
       [<matplotlib.axes._subplots.AxesSubplot object at 0x7fa2a291f518>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7fa2a28d4668>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7fa2a28869e8>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7fa2a2838d68>]],
      dtype=object)
```



5.6 Data Processing

```
[ ]
    #Seperate data
    y =data.target.values
    x1=data.drop(["target"],axis=1)
```

```
[ ]  #Normalization
     x = (x1 - np.min(x1))/(np.max(x1)-np.min(x1)).values
```

CHAPTER 6

MODEL BUILDING AND EVALUATION

In this project, I took 4 algorithms and varied their various parameters and compared the final models. I split the dataset into 80% training data and 20% testing data.

Next, I'll import all the Machine Learning algorithms I will be using.

1.K Neighbors Classifier

2.Logistic Regression Classifier

3.Decision Tree Classifier

4.Support Vector Machine Python

6.1 K NEIGHBORS CLASSIFIER

KNN (K — Nearest Neighbors) is one of many (supervised learning) algorithms used in data mining and machine learning, it’s a classifier algorithm where the learning is based “how similar” is a data (a vector) from other.

Following steps are implemented for K-NN algorithm:

1 — Receive an unclassified data;

2 — Measure the distance (Euclidian, Manhattan, Minkowski or Weighted) from the new data to all others data that is already classified;

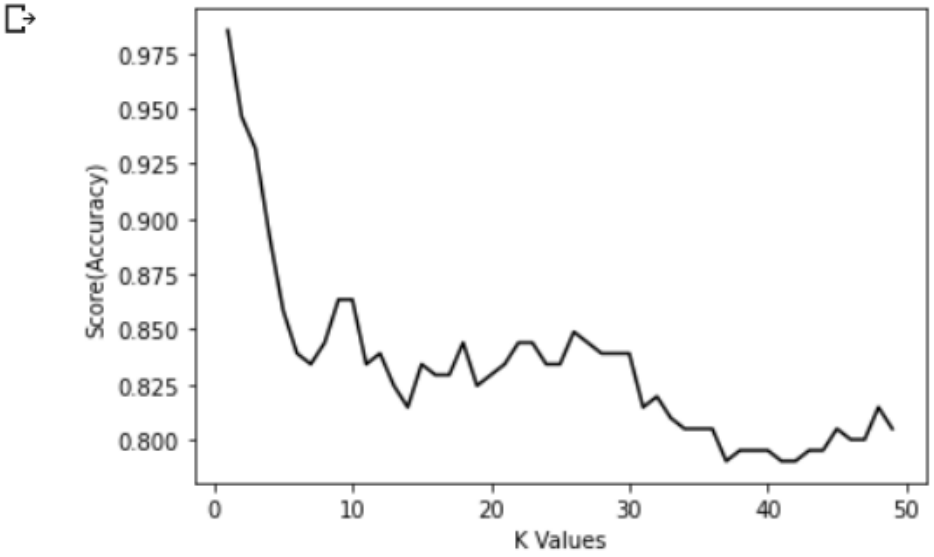
3 — Gets the K(K is a parameter that you difine) smaller distances;

4 — Check the list of classes had the shortest distance and count the amount of each class that appears;

5 — Takes as correct class the class that appeared the most times;

6 —Classifies the new data with the class that you took in step 5;

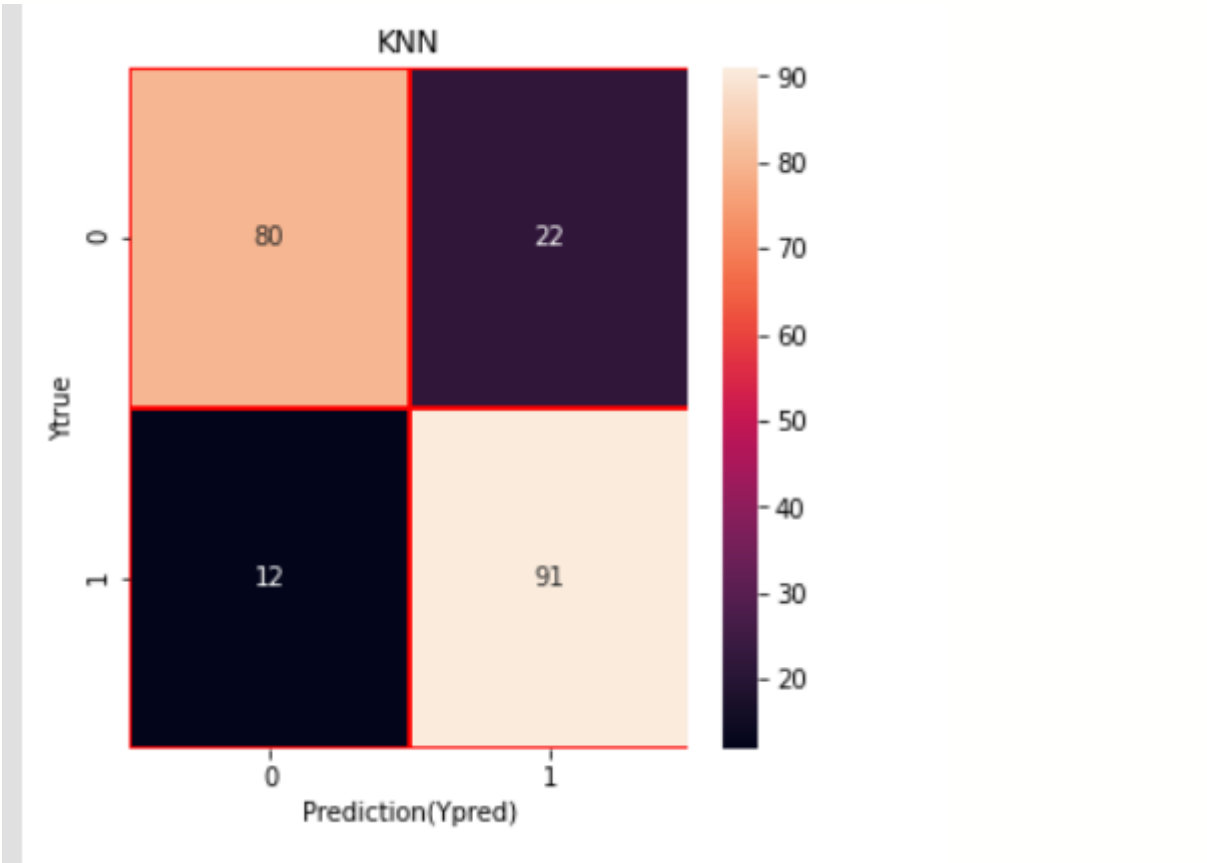
```
[ ]  #Create-KNN-model
     from sklearn.neighbors import KNeighborsClassifier
     #Find Optimum K value
     scores = []
     for each in range(1,50):
         KNNfind = KNeighborsClassifier(n_neighbors = each)
         KNNfind.fit(xtrain.T,ytrain.T)
         scores.append(KNNfind.score(xtest.T,ytest.T))

     plt.plot(range(1,50),scores,color="black")
     plt.xlabel("K Values")
     plt.ylabel("Score(Accuracy)")
     plt.show()

     KNNfind = KNeighborsClassifier(n_neighbors = 24) #n_neighbors = K value
     KNNfind.fit(xtrain.T,ytrain.T) #learning model
     prediction = KNNfind.predict(xtest.T)
     print("{}-NN Score: {}".format(25,KNNfind.score(xtest.T,ytest.T)))
     KNNscore = KNNfind.score(xtest.T,ytest.T)
```




```
25-NN Score: 0.8341463414634146
```

```
[ ]  #Confusion Matrix

     yprediciton2= KNNfind.predict(xtest.T)
     ytrue = ytest

     from sklearn.metrics import confusion_matrix
     CM = confusion_matrix(ytrue,yprediciton2)
```

```
[ ]  #CM visualization

     import seaborn as sns
     import matplotlib.pyplot as plt

     f, ax = plt.subplots(figsize=(5,5))
     sns.heatmap(CM,annot = True, linewidths=0.5,linecolor="red",fmt=".0f",ax=ax)
     plt.xlabel("Prediction(Ypred)")
     plt.ylabel("Ytrue")
     plt.title('KNN')
     plt.show()
```



We have achieved the accuracy of 83 % when k=25

6.2 Logistic Regression Classifier

Logistic Regression was used in the biological sciences in early twentieth century. It was then used in many social science applications.

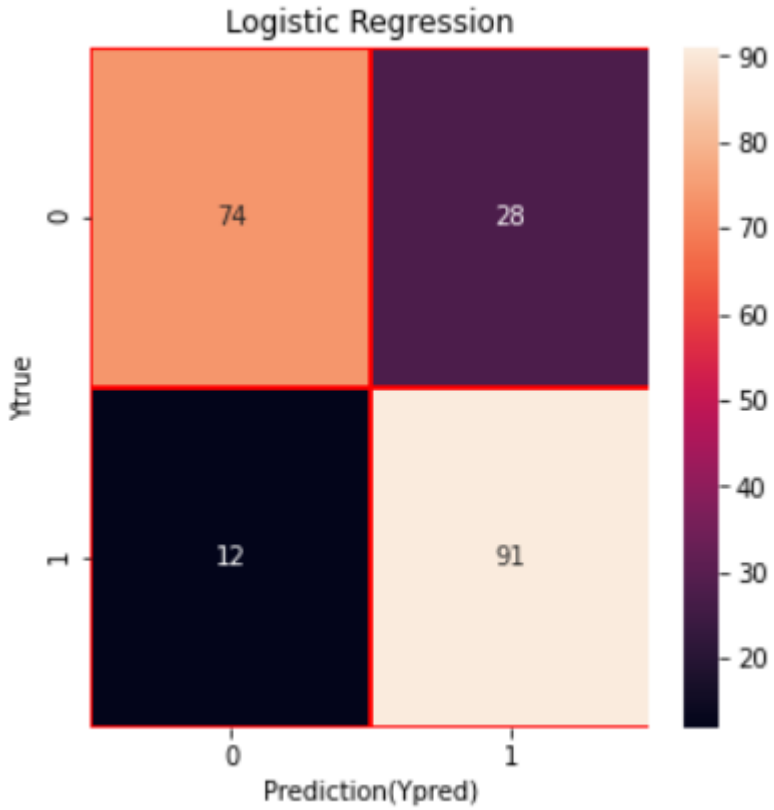
Logistic Regression Algorithm

```
[ ]  #LR with sklearn
     from sklearn.linear_model import LogisticRegression
     LR = LogisticRegression()
     LR.fit(xtrain.T,ytrain.T)
     print("Test Accuracy {}".format(LR.score(xtest.T,ytest.T)))
     LRscore =LR.score(xtest.T,ytest.T)
```

```
Test Accuracy 0.8048780487804879
```

```
[ ]  #Confusion Matrix

     yprediciton1= LR.predict(xtest.T)
     ytrue = ytest

     from sklearn.metrics import confusion_matrix
     CM = confusion_matrix(ytrue,yprediciton1)
```

```
[ ]  #CM visualization

     import seaborn as sns
     import matplotlib.pyplot as plt

     f, ax = plt.subplots(figsize=(5,5))
     sns.heatmap(CM,annot = True, linewidths=0.5,linecolor="red",fmt=".0f",ax=ax)
     plt.xlabel("Prediction(Ypred)")
     plt.ylabel("Ytrue")
     plt.title('Logistic Regression')
     plt.show()
```



We have achieved the accuracy of 80%

6.3 Decision Tree Classifier

Decision Tree algorithm belongs to the family of supervised learning algorithms. Unlike other supervised learning algorithms, the decision tree algorithm can be used for solving regression and classification problems too.

ADVANTAGES

- Decision trees generate understandable rules.
- Decision trees perform classification without requiring much computation.
- Decision trees are capable of handling both continuous and categorical variables.

DISADVANTAGES

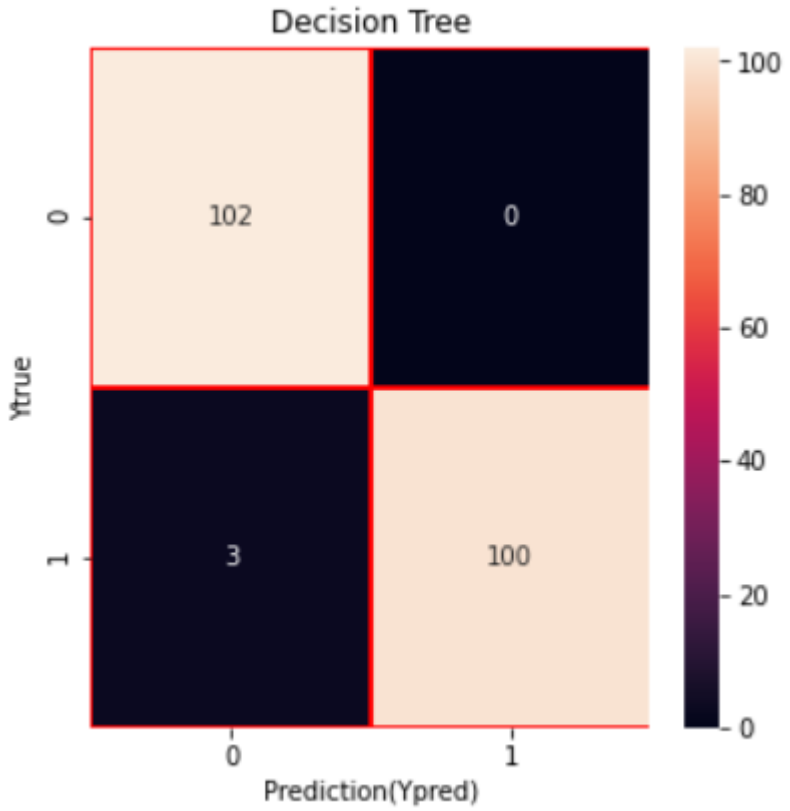
- Decision trees are less appropriate for estimation tasks where the goal is to predict the value of a continuous attribute.
- Decision trees are prone to errors in classification problems with many class and a relatively small number of training examples.

```
#Decision Tree Algorithm

from sklearn.tree import DecisionTreeClassifier
DTC = DecisionTreeClassifier(random_state=2)
DTC.fit(xtrain.T,ytrain.T) #learning
#prediciton
print("Decision Tree Score: ",DTC.score(xtest.T,ytest.T))
DTCscore = DTC.score(xtest.T,ytest.T)
```

```
Decision Tree Score:  0.9853658536585366
```

```
[ ]
    #Confusion Matrix

    yprediciton2= DTC.predict(xtest.T)
    ytrue = ytest

    from sklearn.metrics import confusion_matrix
    CM = confusion_matrix(ytrue,yprediciton2)
```

```
[ ]  #CM visualization

    import seaborn as sns
    import matplotlib.pyplot as plt

    f, ax = plt.subplots(figsize=(5,5))
    sns.heatmap(CM,annot = True, linewidths=0.5,linecolor="red",fmt=".0f",ax=ax)
    plt.xlabel("Prediction(Ypred)")
    plt.ylabel("Ytrue")
    plt.title('Decision Tree')
    plt.show()
```



-

We have achieved the accuracy of 98%

6.4 Support Vector Machines

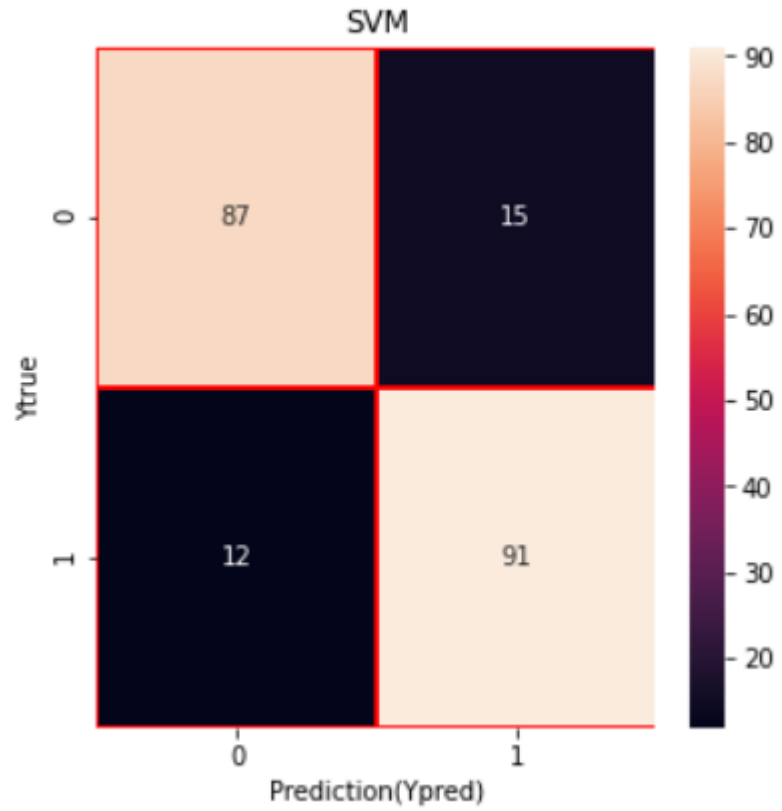
Machine learning involves predicting and classifying data and to do so we employ various machine learning algorithms according to the dataset.

- Explanation of support vector machine (SVM), a popular machine learning algorithm or classification
- Implementation of SVM in R and Python
- Learn about the pros and cons of Support Vector Machines (SVM) and its different applications

```
#SVM with Sklearn

from sklearn.svm import SVC

SVM = SVC(random_state=42)
SVM.fit(xtrain.T,ytrain.T)  #learning
#SVM Test
print ("SVM Accuracy:", SVM.score(xtest.T,ytest.T))

SVMscore = SVM.score(xtest.T,ytest.T)
```

```
SVM Accuracy: 0.8682926829268293
```

```
#Confusion Matrix

yprediciton2= SVM.predict(xtest.T)
ytrue = ytest

from sklearn.metrics import confusion_matrix
CM = confusion_matrix(ytrue,yprediciton2)

#CM visualization

import seaborn as sns
import matplotlib.pyplot as plt

f, ax = plt.subplots(figsize=(5,5))
sns.heatmap(CM,annot = True, linewidths=0.5,linecolor="red",fmt=".0f",ax=ax)
plt.xlabel("Prediction(Ypred)")
plt.ylabel("Ytrue")
plt.title('SVM')
plt.show()
```



We have achieved the accuracy of 86%

6.5 Comparison of algorithms

From the above results we can say that Decision tree classifier model works better for predicting the occurrence of heart dataset based on our given input data file with the accuracy nearly 98% for the test data set.

CHAPTER 7

CONCLUSION

In this project various Machine Learning algorithms such as, K Neighbors Classifier, Support Vector Classifier, Decision Tree Classifier and Logistic Regression are used. Parameters are varied across each model to improve their scores.

Following are the scores achieved for various algorithms.

1.K Neighbors Classifier: 83.41%

2.Support Vector Classifier: 86.82%

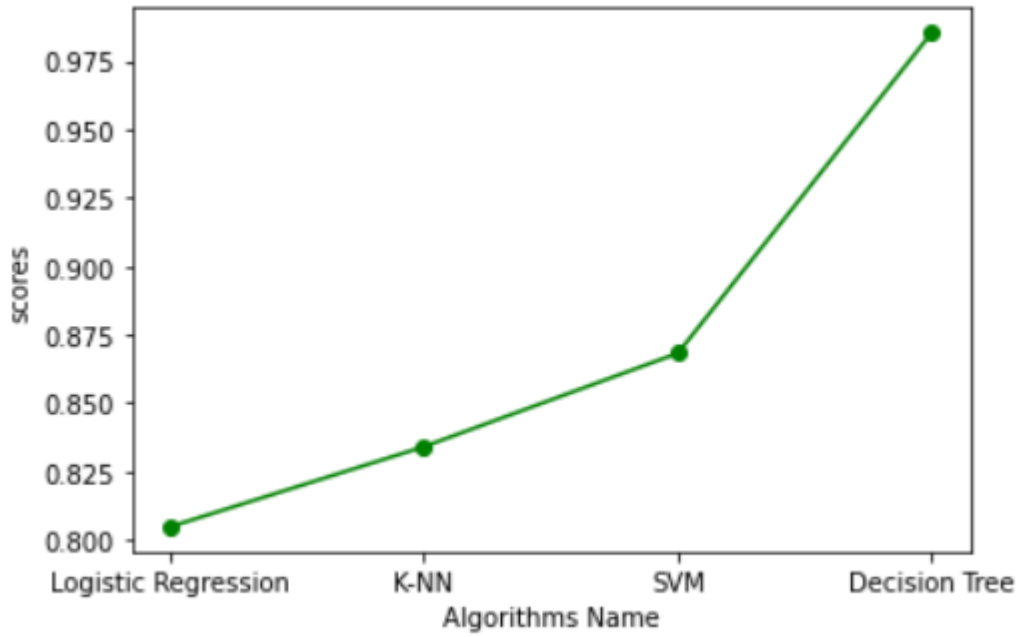
3.Decision Tree Classifier: 98.53%

4.Logistic Regression Classifier: 80.48%

Comparision of algorithms

```
[ ]  #comparison

     scores=[LRscore,KNNscore,SVMscore,DTCscore]
     AlgorthmsName=["Logistic Regression","K-NN","SVM","Decision Tree",]
```

```
[ ]  #create plot

     plt.plot(AlgorthmsName,scores,color='green', marker='o')
     plt.xlabel('Algorithms Name')
     plt.ylabel('scores')
     plt.show()
```



From the above graph we can say that decision tree is having highest score when compare to logistic regression,k-NN,SVM algorithms

In conclusion Decision Tree Classifier achieved the highest score of 98. So Decision Tree Classifier model is recommended to be used for better heart disease prediction.

REFERENCES:

1. https://en.wikipedia.org/wiki/Machine_learning
2. https://www.edureka.co/blog/what-is-data-science/
3. https://www.edureka.co/blog/data-science-applications/
4. https://www.kaggle.com/johnsmith88/heart-disease-dataset