GOPALAN COLLEGE OF
ENGINEERING AND MANAGEMENT

DEPARTMENT OF CSE

Presentation on Internship

ECONOMIC ANALYSIS:

World GDP Prediction

Contents

TABLE OF CONTENTS

CHAPTER:1

INTRODUCTION

Python For Data Science

CHAPTERS	CONTENTS	PG. NOS
	ABSTRACT	V

CHAPTER:1

INTRODUCTION TO DATA SCIENCE

What is data science?

Data science combines math and statistics, specialized programming, advanced analytics, artificial intelligence (AI), and machine learning with specific subject matter expertise to uncover actionable insights hidden in an organization's data. These insights can be used to guide decision making and strategic planning

Needs of data science

Data science is an emerging field of science that has multiple aspects – for one, it studies and examines a huge amount of data; for another, its branches extend in almost every field.

The data we work on is not simple; it is complex data that is structured in many layers. Data science is founded on three main components, and they are statistics, mathematics, and programming language.

Artificial intelligence encapsulates the concepts of all three fields and acts as the machinery or brain of data science. Data science uses techniques, procedures, algorithms, rules, and tools from all these three components and works as a unified mechanism to solve the complex problems that arise in the world around us

Data science in industries

Data science is now more important than ever. The reason for this is data transformation. In the past, the data was in a structured format, was compact, and could be processed by straightforward BI tools.

However, most data nowadays is either semi-structured or unstructured, meaning it takes the form of multimedia such as photos, audio, and videos. Therefore, sophisticated analytical methods that can handle vast volumes of such heterogeneous data are needed for this data.

Data science in job roles

A Data Scientist is a professional who collects large amounts of data using analytical, statistical, and programmable skills. It is their responsibility to use data to develop solutions tailored to meet the organization unique needs.

Phases of Data Science:

***This does not mean that there is not a common workflow to most of these projects however. In the majority of cases, a Data Science project will have to go through five key stages: defining a problem, data processing, modelling, evaluation and deployment*.**

Python of Data science:

Python is a programming language widely used by Data Scientists.

Python has in-built mathematical libraries and functions, makingit easier to calculate mathematical problems and to perform dataanalysis.

Basics of Python:

Python is a popular programming language. It was created byGuido van Rossum, and released in 1991.

It is used for:

- **web development (server-side), software development, mathematics, system scripting.**

CONDITIONS AND LOOPS

Python has two forms of loops: for loop and while loop. Most generic form of loop, that checks whether the condition is true at the start of each iteration. Expects the condition to become false at some point during the iterations of the loop. If condition is never changed, this creates an 'infinite' loop. In Python programming, Operators in general are used to perform operations on values and variables. These are standard symbols used for the purpose of logical and arithmetic operations. In this article, we will look into different types of Python operators.

OPERATORS: These are the special symbols. Eg- + , * , /, etc.

OPERAND: It is the value on which the operator is applied.

Types of Operators in Python

- ***Arithmetic Operators***
- ***Comparison Operators***
- ***Logical Operators***
- ***Bitwise Operators***
- ***Assignment Operators***
- ***Identity Operators and Membership Operators***

Arithmetic Operators in Python

Python Arithmetic operators are used to perform basic mathematical operations like addition, subtraction, multiplication, and division.

FUNCTIONS

Python Functions is a block of statements that return the specific task. The idea is to put some commonly or repeatedly done tasks together and make a function so that instead of writing the same code again and again for different inputs, we can do the function calls to reuse code contained in it over and over again.

Some Benefits of Using Functions

- ***Increase Code Readability***
- ***Increase Code Reusability***

ADVANTAGES AND DISADVANTAGES OF PYTHON

1. Easy to Read, Learn and Write

Python is a high-level programming language that has English-like syntax. This makes it easier to read and understand the code.

Python is really easy to pick up and learn, that is why a lot of people recommend Python to beginners. You need less lines of code to perform the same task as compared to other major languages like C/C++ and Java.

2. Improved Productivity

Python is a very productive language. Due to the simplicity of Python, developers can focus on solving the problem. They don't need to spend too much time in understanding the syntax or behaviour of the programming language. You write less code and get more things done.

3. Interpreted Language

Python is an interpreted language which means that Python directly executes the code line by line. In case of any error, it stops further execution and reports back the error which has occurred.
Python shows only one error even if the program has multiple errors. This makes debugging easier.

4. Dynamically Typed
Python doesn't know the type of variable until we run the code. It automatically assigns the data type during execution. The programmer doesn't need to worry about declaring variables and their data types.
5. Free and Open-Source
Python comes under the OSI approved open-source license. This makes it free to use and distribute. You can download the source code, modify it and even distribute your version of Python. This is useful for organizations that want to modify some specific behavior and use their version for development.

6. Vast Libraries Support
The standard library of Python is huge, you can find almost all the functions needed for your task. So, you don't have to depend on external libraries.

7. Portability
In many languages like C/C++, you need to change your code to run the program on different platforms.
That is not the same with Python. You only write once and run it anywhere.
However, you should be careful not to include any system-dependent features.

1. Slow Speed
We discussed above that Python is an interpreted language and dynamically typed language. The line- by-line execution of code often leads to slow execution.

2. Not Memory Efficient
To provide simplicity to the developer, Python has to do a little trade-off. The Python programming language uses a large amount of memory. This can be a disadvantage while building applications when we prefer memory optimization.

3. Weak in Mobile Computing
Python is generally used in server-side programming. We don't get to see Python on the client-side or mobile applications because of the following reasons. Python is not memory efficient, and it has slow processing power as compared to other languages.

4. Database Access
Programming in Python is easy and stress-free. But when we are interacting with the database, it lacks behind. The Python's database access layer is primitive and underdeveloped in comparison to the popular technologies like JDBC and ODBC.

5. Runtime Errors
As we know Python is a dynamically typed language so the data type of a variable can change anytime. A variable containing integer number may hold a string in the future, which can lead to Runtime Errors

Numpy Array:

Python lists are a substitute for arrays, but they fail to deliver the performance required while computing large sets of numerical data. To address this issue, we use a Python library called NumPy. The word NumPy stands for Numerical Python. NumPy

offers an array object called numpy array. They are similar to standard Python sequences but differ in certain key factor Shape,Dimension size data type in Numpy:

The number of dimensions and items in an array is defined by its shape, which is a tuple of N non-negative integers that specify the sizes of each dimension. The type of items in the array is specified by a separate data-type object (type), one of which is associated with each array.

Pandas:

pandas is a Python package providing fast, flexible, and expressive data structures designed to make working with "relational" or "labelled" data both easy and intuitive. It aims to be the fundamental high-level building block for doing practical, real-world data analysis in Python.

DataFrame:

Data Frame is a 2-dimensional labeled data structure with columns of potentially different types. You can think of it like a spreadsheet or SQL table, or a dict of Series objects. It is generally the most used panda's object.

A Pandas Data Frame is a 2-dimensional data structure, like a 2 dimensional array, or a table with rows and columns.

Datavisualization:

In today's world, a lot of data is being generated daily. And sometimes to analyses this data for certain trends, patterns may become difficult if the data is in its raw format. To overcome this data visualization comes into play. Data visualization provides a good, organized pictorial representation of the data which makes it easier to understand, observe, analyse. In this tutorial, we will discuss how to visualize datausing Python

MATPLOTLIB

Matplotlib is a cross-platform, data visualization and graphical plotting library (histograms, scatter plots, bar charts, etc) for Python and its numerical extension NumPy. As such, it offers a viable open-source alternative to MATLAB. Developers can also use matplotlib's APIs (Application Programming Interfaces) to embed plots in GUI applications.

The plot () function is used to draw points (markers) in a diagram. By default, the plot() function draws a line from point to point. The function takes parameters for specifying points in the diagram. Parameter 1 is an array containing the points on the x-axis.

SEABORN

Python Seaborn library is a widely popular data visualization library that is commonly used for data science and machine learning tasks. You build it on top of the matplotlib data visualization library and can perform exploratory analysis. You can create interactive plots to answer questions about your data. Set aspects of the visual theme for all matplotlib and seaborn plots. This function changes the global defaults for all plots using the matplotlib rc Params syste

LINEAR REGRESSION

Regression searches for relationships among variables. For example, you can observe several employees of some company and try to understand how their salaries depend on their features, such as experience, education level, role, city of employment, and so on

8.1. Preliminaries. import pandas as pd con = pd. ...

8.2. Linear regression with a single explanatory variable. ...

8.3. Regression diagnostics. ...

8.4. Histogram of residuals. ...

8.5. Boxplot of residuals. ...

8.6. Q-Q plot. ...

8.7. Fit plot. ...

8.8. Fit plot in seaborn.

SUPPORT VECTOR

Support vector machines (SVMs) are a set of supervised learning methods used for classification, regression and outliers' detection. The advantages of support vector machines are: Effective in high dimensional spaces. Still effective in cases where number of dimensions is greater than the number of samples.

The SVM kernel is a function that takes low-dimensional input space and transforms it into higherdimensional space, ie it converts nonseparable problems to separable problems. It is mostly useful in nonlinear separation problems

VECTOR

A vector in a simple term can be considered as a single-dimensional array. With respect to Python, a vector is a one-dimensional array of lists. It occupies the elements in a similar manner as that of a Python list.

We can represent a vector in Python as a NumPy array. A NumPy array can be created from a list of numbers. For example, below we define a vector with the length of 3 and the integer values 1, 2 and 3. The example defines a vector with 3 elements.

LOGISTIC REGRESSION

Logistic Regression in Python With scikit-learn.

- ***Import packages, functions, and classes.***
- ***Get data to work with and, if appropriate, transform it.***
- ***Create a classification model and train (or fit) it with your existing data.***
- ***Evaluate your model to see if its performance is satisfactory.***

Binary logistic regression: In this approach, the response or dependent variable is dichotomous in nature—i.e., it has only two possible outcomes (e.g. 0 or 1). Some popular examples of its use include predicting if an e-mail is spam or not spam or if a tumor is malignant or not malignant. Sk-learn pipeline:

The Scikit-learn pipeline is a tool that chains all steps of the workflow together for a more streamlined procedure. The key benefit of building a pipeline is improved readability.
Pipelines can execute a series of transformations with one call, allowing users to attain results with less code.

In software, a pipeline means performing multiple operations (e.g., calling function after function) in a sequence, for each element of an inerrable, in such a way that the output of each element is the input of the next. In Python, you can build pipelines in various ways, some simpler than others.

Data science is a broad and interdisciplinary field that uses scientific methods, processes, algorithms, and systems to extract knowledge and insights from data. Data science topics cover various aspects of data analysis, such as data collection, data manipulation, data visualization, data mining, machine learning, statistics, and more. Here are some of the best data science topics that you can learn and explore:

- ***Data Mining: Data mining is the process of discovering patterns and trends in large and complex data sets. Data mining techniques include classification, clustering, association rules, anomaly detection, and more. Data mining can help you find useful information and insights from data, such as customer behavior, market segmentation, fraud detection, and more. Data mining is one of the core topics in the data science syllabus[1].***
- ***Data Visualization: Data visualization is the presentation of data in a graphical or visual format. Data visualization can help you communicate data and analytics effectively, as well as explore and understand data better. Data visualization tools and techniques include charts, graphs, maps, dashboards, and more. Data visualization is another essential topic in data science that covers the understanding and use of basic and advanced types of graphs[2].***
- ***Machine Learning: Machine learning is the branch of data science that deals with creating and applying algorithms and models that can learn from data and make predictions or decisions. Machine learning can help you solve complex problems and tasks that are difficult or impossible to do with traditional programming. Machine learning topics include supervised and unsupervised learning, regression, classification, clustering, neural networks, deep learning, natural language processing, computer vision, and more. Machine learning is one of the most popular and in-demand topics in data science[3].***

CHAPTER : 2

ABSTRACT

The significance of GDP may be observed in the fact that it gives information on the size and performance of an economy. The rate of growth of real GDP is widely used as an indicator of the overall health of the economy. In general, a growth in real GDP is viewed as a sign that the economy is performing well. This project aims is to predict the world's GDP by using machine learning and also calculate their year to year growth .Every year to year GDP is fluctuating it is important to know the current and importance of the factors which is affected for the country's GDP. By showing this thing in a graph it is easy to understand. This research based project also calculating the importance of features affected for the calculation of the GDP. It is very useful for the viewer to view the GDP growth of the different countries and also they can see the best and worst predict performance.

INTRODUCTION

Gross Domestic Product is the total monetary value of the all the finished goods and services produced within a country in a specific time period.Countries with higher GDPs will have a greater emphasis on the labour and goods produced inside their borders, and will, on average, have a higher standard of life. The gross domestic product is usually calculated on an annual basis, although it may also be calculated quarterly. Typically the Gross Domestic Product or service (GDP) is vital since it provides information on the type and performance of an economy. Typically the project "Global GROSS DOMESTIC PRODUCT Prediction" is structured on research. Typically the goal of this project is to forecast the GROSS DOMESTIC PRODUCT per capita of various countries, as well as being the relevance of the factors that influence GROSS DOMESTIC PRODUCT calculation. We can achieve the best and worst prediction performance by evaluating four different learning regressions (Linear Regression, SVM, Random Forest, and Gradient Boosting) in this study. Our research's main objective is to use Machine Learning and Python to anticipate GDP growth. In this work we have used the library numPy for working with arrays, pandas used to perform data analysis and manipulation, and the matplotlib library for plot the points. We used four different learning regressions: linear regression, support vector machine, random forest, and gradient boosting.

CHAPTER 3

LITERATURE SURVEY

World's GDP prediction" is a researched based project. It is important to everyone to know about their country's GDP. By using some algorithm here I calculated importance of features and how much they affect in calculation of GDP and in the next few what will be our and other countries economic growth. For doing these I referred some papers to gain more knowledge about this. In this paper author explain about the GDP from the basic points and their types, ways of calculating GDP, advantages and disadvantages. This research describes regression and provides comprehensive information on linear regression, simple and multiple linear regression, implementing linear regression in Python, python libraries for linear regression, and simple and multiple linear regression with scikit-learn.In this Paper they gone through about support vector machine, what is SVM, how that algorithm works and some information about support vector regression, implementing SVR in python which includes importing libraries, reading the dataset and feature scaling.In this study author explains about the Random forest algorithm, how to import libraries and dataset, how that dataset splits, how that predict the results.The facts regarding gradient boosting, the genesis of boosting from learning theory, and adaboost will be provided in this work. The loss function, weak learners, and the additive model all play a role in how gradient boosting works. How to use multiple regularisation strategies to increase performance over the base algorithm.This research looks at how machine learning may be used to forecast economic variables. Although AI has been integrated into economics, machine learning projections have yet to be completely tested. In the instance of current cases from G7 countries, a comparison of employing AR models and machine learning to anticipate GDP and consumer price is undertaken. This paper joins a developing writing that assesses the general achievement of ML models in estimating throughout the more conventional time-series procedures. In this paper, we examine the presentation of various ML calculations in acquiring precise now projects of genuine total national output (GDP) development for New Zealand. We utilize various vintages of authentic GDP information and numerous vintages of an enormous highlights set. Our goal is to use multiple sophisticated machine learning techniques to develop and evaluate a forecasting model for GDP changes. We investigate if machine learning algorithms can increase forecasting accuracy and attempt to identify the factors that drive economic recovery and predict recession. In this paper, we explore the exhibition of various ML calculations in getting exact now projects of the current quarter genuine (GDP) development for New Zealand. We utilize various vintages of recorded GDP information and different vintages of an enormous highlights set - involving roughly 550 homegrown and worldwide factors - to assess the continuous exhibition of these calculations over the 2009-2018 period. The primary goal of this article is to examine the trajectory of GDP to determine if there is any change in the trend between the years 1981-82 to 1990-91 and 1991-92 to 2001-2002, i.e. two decades before and after the liberalisation programme began. 2. Investigate the nature of the difference, if one exists. 3. Determine the reasons for the difference. 4. Create a GDP forecasting model. In this paper it will provide the information about India's economy, what are all the factors affecting, how we can calculate the GDP, how we can predict the data.This paper give the information about the projected real GDP growth, relative income analysis, scenario analysis, economic growth in 2050

Data source:

The following table shows the list of attributes on which we are working.

Table 2.1: List of Attributes

S no	Attribute Name	Description
1	Country	Name of the Country
2	Region	Continent of the Country(sub-anomical)
3	Population	Number of People
4	Area	Sq. Mi of country
5	Pop.Density	Pop Density per sq mi.
6	Coastline	Coast/area ratio
7	Net Migration	Migration Ratio
8	Infant Mortality	Per 1000 births
9	Gdp	$ per captia
10	Litreacy(%)	Percent of country of litreacy
11	Phones	Per 1000
12	Arable(%)	Percent of air in crops
13	Crops(%)	Percent of crops per acre
14	Other(%)	Percent of other factors

CHAPTER 3 AIM AND SCOPE OF PRESENT INVESTIGATION

3.1 EXISTING SYSTEM:

GDP are often made based on Analytical‘ intuition and experience rather than on the knowledge rich data hidden in the database. This practice leads to unwanted biases, errors and excessivel costs which affects the quality of service provided to senontical’. GDP are a serious risk to our world economic progress.We can put an end to GDP miscalculations by informing the public and filing claims and suits against the Analysts and Economic whore downs.

Disadvantages:

- Prediction is not possible at early stages.
- In the Existing system, practical use of collected data is time consuming.
- Any faults occurred by the doctor or hospital staff in predicting would lead to fatal incidents.
- Highly expensive and laborious process needs to be performed before treating the patient to find out if he/she has any chances to get heart disease in future.

3.2 PROPOSED SYSTEM:

In this researched based GDP predictor we have collect the one dataset which has almost all the country's factors value for the calculation of the GDP. In that we are selecting some of the country's name and showing that country's GDP results in graphs and also showing the importance of factors, And using 4 linear regressors we are calculating Mean Absolute Error, Root mean squared error (RMSE) and R-squared Score (R2_Score) and shows that result in graph

3.2 FEASIBILITY STUDY:

3.2.1 Economic Feasibility:

It is defined as the process of assessing the benefits and costs associated with the development of project. A proposed system, which is both operationally and technically feasible, must be a good investment for the organization. With the proposed system the users are greatly benefited as the users can be able to detect the fake news from the real news and are aware of most real and most fake news published in the recent years. This proposed system does not need any additional software and high system configuration. Hence the proposed system is economically feasible.

3.2.2 Technical Feasibility:

The technical feasibility infers whether the proposed system can be developed considering the technical issues like availability of the necessary technology, technical capacity, adequate response and extensibility. The project is decided to build using Python. Jupyter Notebook is designed for use in distributed environment of the internet and for the professional programmer it is easy to learn and use effectively. As the developing organization has all the resources available to build the system therefore the proposed system is technically feasible.

3.2.3 Operational Feasibility:

Operational feasibility is defined as the process of assessing the degree to which a proposed system solves business problems or takes advantage of business opportunities. The system is self-explanatory and doesn‘t need any extra sophisticated training. The system has built-in methods and classes which are required to produce the result. The application can be handled very easily with a novice user. The overall time that a user needs to get trained is 14 less than one hour. As the software that is used for developing this application is very economical and is readily available in the market. Therefore, the proposed system is operationally feasible.

3.4 EFFORT, DURATION AND COST ESTIMATION USING COCOMO MODEL:

The COCOMO (Constructive Cost Model) model is the most complete and

thoroughly documented model used in effort estimation. The model provides detailed formulas for determining the development time schedule, overall development effort, and effort breakdown by phase and activity as well as maintenance effort. COCOMO estimates the effort in person months of direct labor. The primary effort factor is the number of source lines of code (SLOC) expressed in thousands of delivered source instructions (KDSI). The model is developed in three versions of different level of detail basic, intermediate, and detailed. The overall modeling process takes into account three classes of systems.

3.4.1. Embedded:

This class of system is characterized by tight constraints, changing environment, and unfamiliar surroundings. Projects of the embedded type are model to the company and usually exhibit temporal constraints.

3.4.2. Organic:

This category encompasses all systems that are small relative to project size and team size and have a stable environment, familiar surroundings and relaxed interfaces. These are simple business systems, data processing systems, and small software libraries.

3.4.3. Semidetached:

The software systems falling under this category are a mix of those of organic and embedded in nature. Some examples of software of this class are operating systems, database management system, and inventory management systems class are operating systems, database management system, and inventory management systems.

Table 3.1: Organic, Semidetached and Embedded system values

TYPE OF PRODUCT	A	B	C	D
Organic	2.4	1.02	2.5	0.38
Semi Detached	3.0	1.12	2.5	0.35
Embedded	3.6	1.20	2.5	0.32

For basic COCOMO Effort = a*(KLOC) b

Type =c*(effort)d

For Intermediate and Detailed COCOMO Effort = a * (KLOC) b* EAF (EAF = product of cost drivers).

Intermediate COCOMO model is a refinement of the basic model, which comes in the function of 15 attributes of the product. For each of the attributes the user of the model has to provide a rating using the following six-point scale.

LO (Low) HI (High)

VL (Very Low) VH (Very High)

NM (Normal) XH (Extra High)

The list of attributes is composed of several features of the software and includes product, computer, personal and project attributes as follows.

3.4.4 Product Attributes:

- **Required reliability (RELY):** It is used to express an effect of software faults ranging from slight inconvenience (VL) to loss of life (VH). The nominal value (NM) denotes moderate recoverable losses.
- **Data bytes per DSI (DATA):** The lower rating comes with lower size of a database. Complexity (CPLX): The attribute expresses code complexity again ranging from straight batch code (VL) to real time code with multiple resources scheduling (XH).

3.4.5 Computer Attributes:

- **Execution time (TIME) and memory (STOR) constraints:** This attribute identifies the percentage of computer resources used by the system. NM states that less than 50% is used; 95% is indicated by XH.]
- **Virtual machine volatility (VIRT):** It is used to indicate the frequency of changes made to the hardware, operating system, and overall software environment. More frequent and significant changes are indicated by higher ratings.
- **Development turnaround time (TURN):** This is a time from when a job is submitted until output becomes received. LO indicated a highly interactive environment, VH quantifies a situation when this time is longer than 12 hours.

3.4.6 Personal Attributes:

- Analyst capability (ACAP) and Analyst programmer capability (PCAP).
- This describes skills of the developing team. The higher the skills, the higher the rating.
- Application experience (AEXP), language experience (LEXP), and virtual machine experience (VEXP)
- These are used to quantify the number of experiences in each area by the development team, more experience, higher rating.

3.4.7 Project Attributes:

- **Modern development practices (MODP):** deals with the amount of use of modern software practices such as structural programming and objectoriented approach.

- **Use of software tools (TOOL):** is used to measure a level of sophistication of automated tools used in software development and a degree of integration among the tools being used. Higher rating describes levels in both aspects.
- **Schedule effects (SCED):** concerns the amount of schedule compression (HI or VH), or schedule expansion (LO or VL) of the development schedule in comparison to a nominal (NM) schedule.

Table 3.2: Project Attributes

	VL	LO	NM	HI	VH	XH
RELY	0.75	0.88	1.00	1.15	1.40	
DATA		0.94	1.00	1.08	1.16	
CPLX	0.70	0.85	1.00	1.15	1.30	1.65
TIME			1.00	1.11	1.30	1.66
STOR			1.00	1.06	1.21	1.56
VIRT		0.87	1.00	1.15	1.30	
TURN		0.87	1.00	1.15	1.30	
ACAP	1.46	1.19	1.00	0.86	0.71	
AEXP	1.29	1.29	1.00	0.91	0.82	
PCAP	1.42	1.17	1.00	0.86	0.70	
LEXP	1.14	1.07	1.00	0.95		
VEXP	1.21	1.10	1.00	0.90		

MODP	1.24	1.10	1.00	0.91	0.82	
TOOL	1.24	1.10	1.00	0.91	0.83	
SCED	1.23	1.08	1.00	1.04	1.10	

Our project is an organic system and for intermediate

COCOMO Effort = a* (KLOC) b * EAF

KLOC = 115

For organic system a

= 2.4 b = 1.02 EAF =

product of cost

Driver‘s effort = 2.4 * (0.115) ^ 1.02 * 1.30

= 1.034

Programmer month's Time for Development = C * (Effort) d

= 2.5 * (1.034) ^ 0.38

= 2.71 months

Cost of programmer = Effort * cost of Programmer per month

= 1.034 * 20000

= 20680

Project cost = 20000 + 20680

= 40680

CHAPTER :4

EXPERIMENTAL OR MATERIALS AND METHODS

4.1 INTRODUCTION TO REQUIREMENT SPECIFICATION:

Software Engineering by James F Peters & WitoldPedrycz Headfirst Java by Ka. A Software Requirements Specification (SRS) is a description of software product, program or set of programs that performs a set of functions in a target environment (IEEE Std. 830-1993).

a. **Purpose:**

   The purpose of software requirements specification specifies the intentions and intended audience of the SRS.

b. **Scope**:

   The scope of the SRS identifies the software product to be produced, the capabilities, application, relevant objects etc. We are proposed to implement Passive Aggressive Algorithm which takes the test and trained data set.

c. **Definitions, Acronyms and Abbreviations Software Requirements Specification:**

   It‘s a description of a particular software product, program or set of programs that performs a set of function in target environment.

d. **References:**

   IEEE Std. 830-1993, IEEE Recommended Practice for Software Requirements specifications thy Sierra and Bert Bates.

e. **Overview:**

   The SRS contains the details of process, DFD‘s, functions of the product, user characteristics. The non-functional requirements if any are also specified.

f. **Overall description:**

   The main functions associated with the product are described in this section of SRS. The characteristics of a user of this product are indicated. The assumptions in this section result from interaction with the project stakeholders.

4.2 REQUIREMENT ANALYSIS:

Software Requirement Specification (SRS) is the starting point of the software developing activity. As system grew more complex it became evident that the goal of the entire system cannot be easily comprehended. Hence the need for the requirement phase arose. The software project is initiated by the client needs. The SRS is the means of translating the ideas of the minds of clients (the input) into a formal document (the output of the requirement phase.) Under requirement specification, the focus is on specifying what has been found giving analysis such as representation, specification languages and tools, and checking the specifications are addressed during this activity. The Requirement phases terminates with the production of the validate SRS document. Producing the SRS document is the basic goal of this phase. The purpose of the Software Requirement Specification is to reduce the communication gap between the clients and the developers. Software Requirement Specification is the medium though which the client and user needs are accurately specified. It forms the basis of software development. A good SRS should satisfy all the parties involved in the system.

4.2.1 Product Perspective:

The application is developed in such a way that any future enhancement can be easily implementable. The project is developed in such a way that it requires minimal maintenance. The software used are open source and easy to install. The application developed should be easy to install and use. This is an independent application which can be easily run on to any system which has Python installed and Jupiter Notebook.

4.2.2 Product Features:

The application is developed in a way that ‗heart disease' accuracy is predicted using Random Forest. The dataset is taken from https://www.datacamp.com/community/tutorials/scikit-learn-credit-card. We can compare the accuracy for the implemented algorithms. User characteristics Application is developed in such a way that its users are v Easy to use v Error free 20 v Minimal training or no training v Patient regular monitor Assumption & Dependencies It is considered that the dataset taken fulfils all the requirements.

4.2.3 Domain Requirements:

This document is the only one that describes the requirements of the system. It is meant for the use by the developers and will also be the bases for validating the final heart disease system. Any changes made to the requirements in the future will have to go through a formal change approval process. User Requirements User can decide on the prediction accuracy to decide on which algorithm can be used in realtime predictions. Non-Functional Requirements ÿ Dataset collected should be in the CSV format ÿ. The column values should be numerical values ÿ Training set and test set are stored as CSV files ÿ Error rates can be calculated for prediction algorithms product.

4.2.4 Requirements Efficiency:

Less time for predicting the Heart Disease Reliability: Maturity, fault tolerance and recoverability. Portability: can the software easily be transferred to another environment, including install ability.

4.2.5 Usability:

How easy it is to understand, learn and operate the software system Organizational. Requirements: Do not block some available ports through the windows firewall. Internet connection should be available Implementation Requirements The dataset collection, internet connection to install related libraries. Engineering Standard Requirements User interface is developed in python, which gets input such stock symbol.

4.2.6 Hardware Interfaces:

Ethernet on the AS/400 supports TCP/IP, Advanced Peer-to-Peer Networking (APPN) and advanced program-to-program communications (APPC). ISDN To connect AS/400 to an Integrated Services Digital Network (ISDN) for faster, more accurate data transmission. An ISDN is a public or private digital communications network that can support data, fax, image, and other services over the same physical interface. We can use other protocols on ISDN, such as IDLC and X.25. Software Interfaces Anaconda Navigator and Jupiter Notebook are used.

4.2.7 Operational Requirements:

a. **Economic:** The developed product is economic as it is not required any hardware interface etc. Environmental Statements of fact and assumptions that define the expectations of the system in terms of mission objectives, environment, constraints, and measures of effectiveness and suitability (MOE/MOS). The customers are those that perform the eight primary functions of systems engineering, with special emphasis on the operator as the key customer.

b. **Health and Safety:** The software may be safety critical. If so, there are issues associated with its integrity level. The software may not be safety-critical although it forms part of a safety-critical system.

- There is little point in producing 'perfect' code in some language if hardware and system software (in widest sense) are not reliable. If a computer system is to run software of a high integrity level, then that system should not at the same time accommodate software of a lower integrity level.
- Systems with different requirements for safety levels must be separated. Otherwise, the highest level of integrity required must be applied to all systems in the same environment.

4.3 SYSTEM REQUIREMENTS

4.3.1 Hardware Requirements:

Processor	:	above 500MHz
Ram	:	4GB
Hard Disk	:	4GB
Input device	:	Standard keyboard and Mouse
Output device	:	VGA and High-Resolution Monitor

4.3.2 SOFTWARE REQUIREMENTS:

Operating System	:	Windows 7 or higher
Programming	:	python 3.6 and related libraries
Software	:	Anaconda Navigator, Jupyter Notebook and Google colab

4.4 SOFTWARE DESCRIPTION

4.4.1 Python:

Python is an interpreted high-level programming language for general-purpose programming. 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. Python features a dynamic type of system and automatic memory management. It supports multiple programming paradigms, including object-oriented, imperative, functional and procedural, and has a large and comprehensive standard library. Python interpreters are available for many operating systems. C Python, the reference implementation of Python, is open-source software and has a communitybased development model, as do nearly all its variant implementations. C Python is managed by the non-profit Python Software Foundation.

4.4.2 Pandas

Pandas is an open-source Python Library providing high-performance data manipulation and analysis tool using its powerful data structures. The name Pandas is derived from the word Panel Data – an Econometrics from Multidimensional data. In 2008, developer Wes McKinney started developing pandas when in need of high performance, flexible tool for analysis of data. Prior to Pandas, Python was majorly used for data mining and preparation. It had very little contribution towards data analysis. Pandas solved this problem.

Using Pandas, we can accomplish five typical steps in the processing and analysis of data, regardless of the origin of data — load, prepare, manipulate, model, and analyze. Python with Pandas is used in a wide range of fields including academic and commercial domains including finance, economics, Statistics, analytics, etc.

Key Features of Pandas:

- Fast and efficient Data Frame object with default and customized indexing.
- Tools for loading data into in-memory data objects from different file formats.
- Data alignment and integrated handling of missing data.
- Reshaping and pivoting of date sets.
- Label-based slicing, indexing and subsetting of large data sets.
- Columns from a data structure can be deleted or inserted.
- Group by data for aggregation and transformations.

- High performance merging and joining of data.
- Time Series functionality.

4.4.3 NumPy:

NumPy is a general-purpose array-processing package. It provides a high-performance multidimensional array object, and tools for working with these arrays. It is the fundamental package for scientific computing with Python. It contains various features including these important ones:

- A powerful N-dimensional array object
- Sophisticated (broadcasting) functions
- Tools for integrating C/C++ and Fortran code
- Useful linear algebra, Fourier transform, and random number capabilities 24
- Besides its obvious scientific uses, NumPy can also be used as an efficient multi-dimensional container of generic data. Arbitrary datatypes can be defined using NumPy which allows NumPy to seamlessly and speedily integrate with a wide variety of databases.

4.4.4 Sckit-Learn:

- Simple and efficient tools for data mining and data analysis
- Accessible to everybody, and reusable in various contexts
- Built on NumPy, SciPy, and matplotlib
- Open source, commercially usable - BSD license

4.4.5 Matploit lib:

- Matplotlib is a python library used to create 2D graphs and plots by using python scripts.
- It has a module named pyplot which makes things easy for plotting by providing feature to control line styles, font properties, formatting axes etc.
- It supports a very wide variety of graphs and plots namely - histogram, bar charts, power spectra, error charts etc.

4.4.6 Jupyter Notebook:

- The Jupyter Notebook is an incredibly powerful tool for interactively developing and presenting data science projects.
- A notebook integrates code and its output into a single document that combines visualizations, narrative text, mathematical equations, and other rich media.

- The Jupyter Notebook is an open-source web application that allows you to create and share documents that contain live code, equations, visualizations and narrative text.
- Uses include data cleaning and transformation, numerical simulation, statistical modeling, data visualization, machine learning, and much more.
- The Notebook has support for over 40 programming languages, including Python, R, Julia, and Scala.
- Notebooks can be shared with others using email, Drop box, Git Hub and the Jupyter Notebook.
- Your code can produce rich, interactive output: HTML, images, videos, LATEX, and custom MIME types.
- Leverage big data tools, such as Apache Spark, from Python, R and Scala. Explore that same data with pandas, scikit-learn, ggplot2, Tensor Flow.

4.5 ALGORITHMS

4.5.1 Logistic Regression

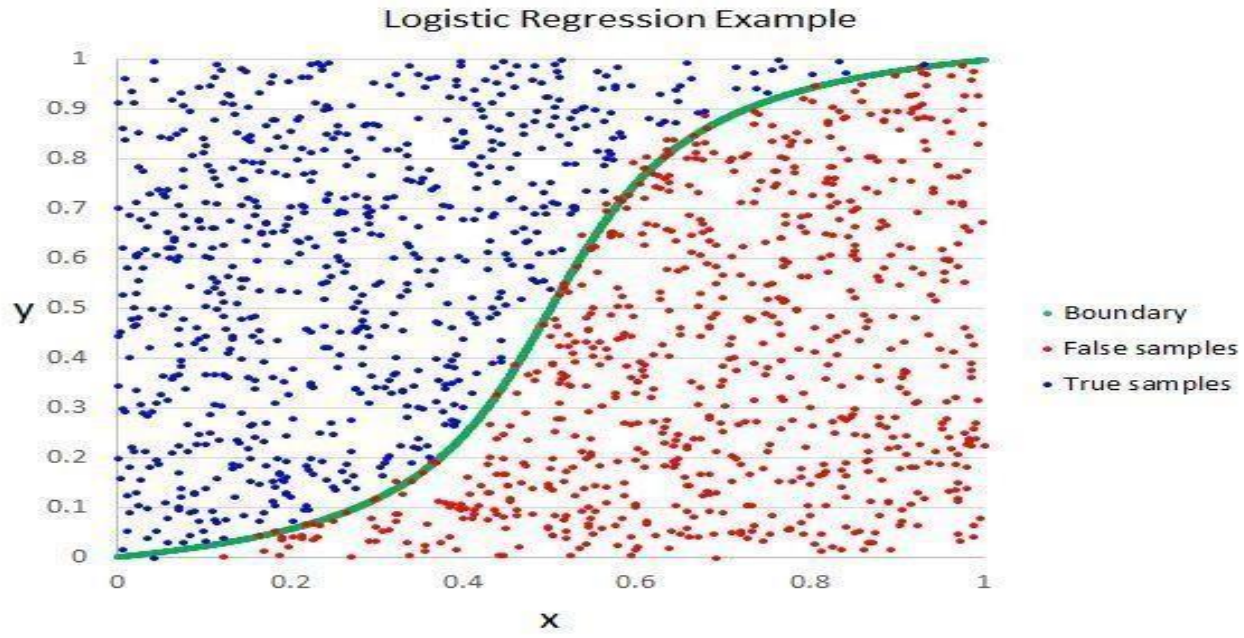
A popular statistical technique to predict binomial outcomes (y = 0 or 1) is Logistic Regression. Logistic regression predicts categorical outcomes (binomial / multinomial values of y). The predictions of Logistic Regression (henceforth, LogR in this article) are in the form of probabilities of an event occurring, i.e., the probability of y=1, given certain values of input variables x. Thus, the results of LogR range between 0-1.

LogR models the data points using the standard logistic function, which is an Sshaped curve also called as sigmoid curve and is given by the equation.

$$f(x) = \frac{1}{1+e^{-(x)}}$$

Logistic Regression Assumptions:

- Logistic regression requires the dependent variable to be binary.
- For a binary regression, the factor level 1 of the dependent variable should represent the desired outcome.
- Only the meaningful variables should be included.
- The independent variables should be independent of each other.
- Logistic regression requires quite large sample sizes.
- Even though, logistic (logit) regression is frequently used for binary variables (2 classes), it can be used for categorical dependent variables with more than 2 classes.
- In this case it‘s called Multinomial Logistic Regression. **Fig 4.5.1: Logistic Regression**



4.5.2 Random Forest:

Random Forest is a supervised learning algorithm which is used for both classification as well as regression. But however, it is mainly used for classification problems. As we know that a forest is made up of trees and more trees means more robust forest.

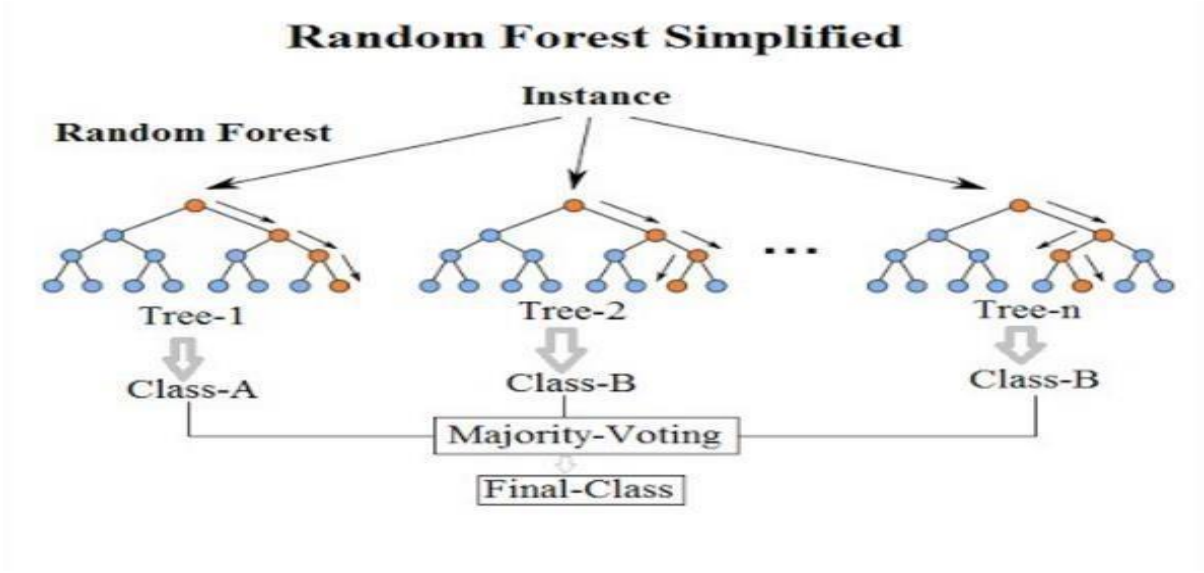
Similarly, random forest creates decision trees on data samples and then gets the prediction from each of them and finally selects the best solution by means of voting. It is ensemble method which is better than a single decision tree because it reduces the over-fitting by averaging the result.

Working of Random Forest with the help of following steps:

- First, start with the selection of random samples from a given dataset.
- Next, this algorithm will construct a decision tree for every sample. Then it will get the prediction result from every decision tree.
- In this step, voting will be performed for every predicted result.
- At last, select the most voted prediction results as the final prediction result.

The following diagram will illustrate its working-

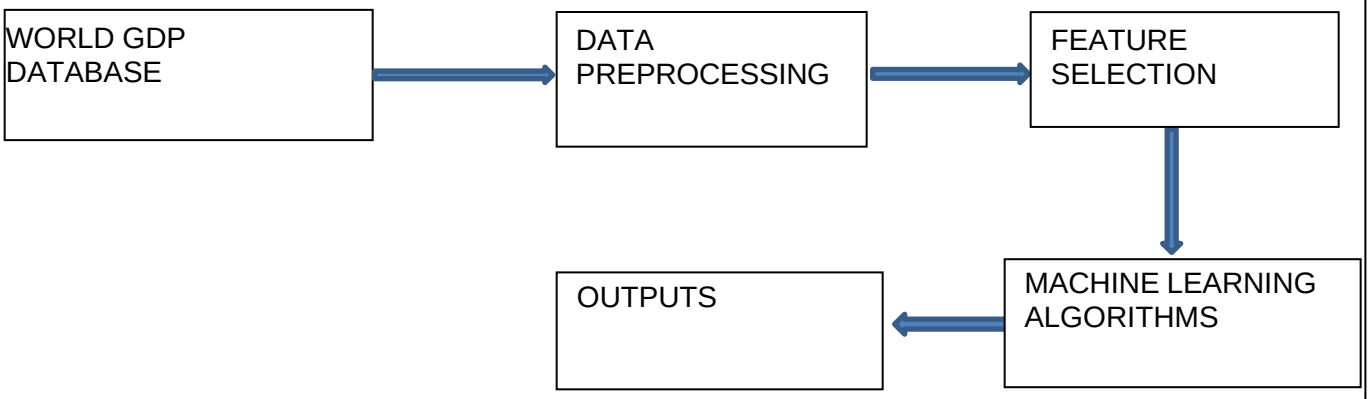
Fig 4.5.2: Random Forest Classifier



4.6 SYSTEM ARCHITECTURE

The below figure shows the process flow diagram or proposed work. First, we collected the World GDP Database from KAGGLE website then preprocessed the dataset and select 16 important features

Fig 4.6: System Architecture



For feature selection we used Recursive feature Elimination Algorithm using Chi2 method and get 16 top features. After that applied ANN and Logistic algorithm individually and compute the accuracy. Finally, we used proposed Ensemble Voting method and compute best method for diagnosis of heart disease.

4.7 MODULES:

The entire work of this project is divided into 4 modules.

They are:

a. **Data Pre-Processing**
b. **Feature**
c. **Classification**
d. **Prediction**

a. Data Pre-processing:

This file contains all the pre-processing functions needed to process all input documents and texts. First, we read the train, test and validation data files then performed some preprocessing like tokenizing, stemming etc. There are some exploratory data analyses is performed like response variable distribution and data quality checks like null or missing values etc.

Data preprocessing is the process of transforming raw data into an understandable format. It is also an important step in data mining as we cannot work with raw data. The quality of the data should be checked before applying machine learning or data mining algorithms.

Preprocessing of data is mainly to check the data quality. The quality can be checked by the following-

- **Accuracy**: To check whether the data entered is correct or not.
- **Completeness**: To check whether the data is available or not recorded.
- **Consistency:** To check whether the same data is kept in all the places that do or do not match.
- **Timeliness**: The data should be updated correctly **Believability**: The data should be trustable.
- **Interpretability**: The understandability of the data.

b. Feature:

Extraction In this file we have performed feature extraction and selection methods from sci-kit learn python libraries. For feature selection, we have used methods like simple bag-of-words and n-grams and then term frequency like tf-idf weighting. We have also used word2vec and POS tagging to extract the features, though POS tagging and word2vec has not been used at this point in the project **Bag of Words:**

It‘s an algorithm that transforms the text into fixed-length vectors. This is possible by counting the number of times the word is present in a document. The word occurrences allow to compare different documents and evaluate their similarities for applications, such as search, document classification, and topic modeling.

The reason for its name, ―Bag-Of-Words‖, is due to the fact that it represents the sentence as a bag of terms. It doesn‘t consider the order and the structure of the words, but it only checks if the words appear in the document.

N-grams:

N-grams are continuous sequences of words or symbols or tokens in a document. In technical terms, they can be defined as the neighbouring sequences of items in a document. They come into play when we deal with text data in NLP(Natural Language Processing) tasks.

TF-IDF Weighting:

TF-IDF stands for term frequency-inverse document frequency and it is a measure, used in the fields of information retrieval (IR) and machine learning, that can quantify the importance or relevance of string representations (words, phrases, lemmas, etc) in a document amongst a collection of documents (also known as a corpus).

c. Classification:

Here we have built all the classifiers for the breast cancer diseases detection. The extracted features are fed into different classifiers. We have used Naive-bayes, Logistic Regression, Linear SVM, Stochastic gradient decent and Random Forest classifiers from sklearn. Each of the extracted features was used in all the classifiers. Once fitting the model, we compared the f1 score and checked the confusion matrix. After fitting all the classifiers, 2 best performing models were selected as candidate models for heart diseases classification. We have performed parameter tuning by implementing GridSearchCV methods on these candidate models and chosen best performing parameters for these classifiers. Finally selected model was used for heart disease detection with the probability of truth. In Addition to this, we have also extracted the top 50 features from our term-frequency tf-idf Vectorizer to see what words are most and important in each of the classes. We have also used PrecisionRecall and learning curves to see how training and test set performs when we increase the amount of data in our classifiers.

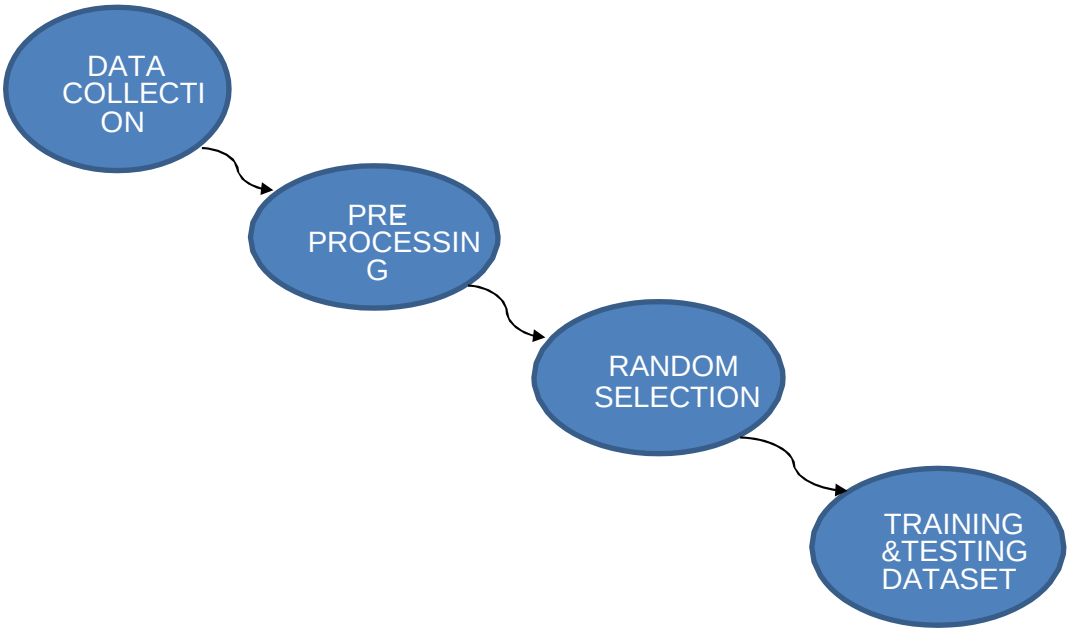
d. Prediction:

Our finally selected and best performing classifier was algorithm which was then saved on disk with name final_model.sav. Once you close this repository, this model will be copied to user's machine and will be used by prediction.py file to classify the heart diseases. It takes a news article as input from user then model is used for final classification output that is shown to user along with probability of truth.

4.8 DATA FLOW DIAGRAM:

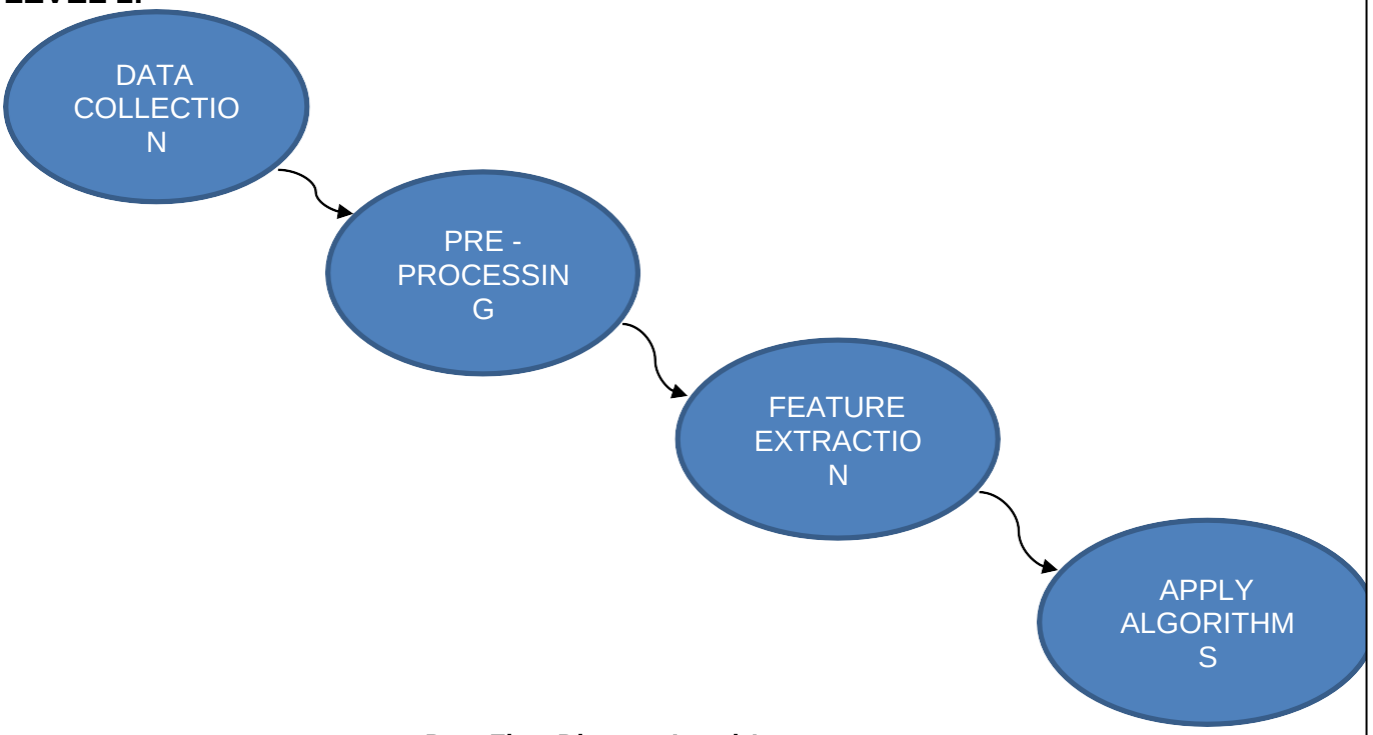
The data flow diagram (DFD) is one of the most important tools used by system analysis. Data flow diagrams are made up of number of symbols, which represents system components. Most data flow modeling methods use four kinds of symbols: Processes, Data stores, Data flows and external entities. These symbols are used to represent four kinds of system components. Circles in DFD represent processes. Data Flow represented by a thin line in the DFD, and each data store has a unique name and square or rectangle represents external entities.

LEVEL:0



DATA FLOW DIAGRAM LEVEL 0

LEVEL 1:



Data Flow Diagram Level 1

CHAPTER 5

RESULT AND DISCUSSION, PERFORMANCE ANALYSIS

In this project, we introduce about the world GDP prediction system with different classifier techniques for the prediction of GDP. The techniques are Random Forest and Logistic Regression: we have analyzed that the Random Forest has better accuracy as compared to Logistic Regression. Our purpose is to improve the performance of the Random Forest by removing unnecessary and irrelevant attributes from the dataset and only picking those that are most informative for the classification task.

1 Load Dataset

```
[ ] df = pd.read_csv("/content/drive/MyDrive/Document from Rajagopal")
```

```
[ ] df.head()
```

	Country	Region	Population	Area (sq. mi.)	Pop. Density (per sq. mi.)	Coastline (coast/area ratio)	Net migration	Infant mortality (per 1000 births)	GDP ($ per capita)	Literacy (%)	Phones (per 1000)	Arable (%)	Crops (%)	Other (%)	Cli
0	Afghanistan	ASIA (EX. NEAR EAST)	31056997	647500	48,0	0,00	23,06	163,07	700.0	36,0	3,2	12,13	0,22	87,65	
1	Albania	EASTERN EUROPE	3581655	28748	124,6	1,26	-4,93	21,52	4500.0	86,5	71,2	21,09	4,42	74,49	
2	Algeria	NORTHERN AFRICA	32930091	2381740	13,8	0,04	-0,39	31	6000.0	70,0	78,1	3,22	0,25	96,53	
3	American Samoa	OCEANIA	57794	199	290,4	58,29	-20,71	9,27	8000.0	97,0	259,5	10	15	75	
4	Andorra	WESTERN EUROPE	71201	468	152,1	0,00	6,6	4,05	19000.0	100,0	497,2	2,22	0	97,78	

```
[ ] df.isnull().sum()
```

```
Country                             0
Region                              0
Population                          0
Area (sq. mi.)                      0
Pop. Density (per sq. mi.)          0
Coastline (coast/area ratio)        0
Net migration                       3
Infant mortality (per 1000 births)  3
GDP ($ per capita)                  1
Literacy (%)                       18
Phones (per 1000)                   4
Arable (%)                          2
Crops (%)                           2
Other (%)                           2
Climate                            22
Birthrate                           3
Deathrate                           4
Agriculture                        15
Industry                           16
Service                            15
dtype: int64
```

Fig description:

```
[ ] df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 227 entries, 0 to 226
Data columns (total 20 columns):
 #   Column                              Non-Null Count  Dtype
---  ------                              --------------  -----
 0   Country                             227 non-null    object
 1   Region                              227 non-null    object
 2   Population                          227 non-null    int64
 3   Area (sq. mi.)                      227 non-null    int64
 4   Pop. Density (per sq. mi.)          227 non-null    object
 5   Coastline (coast/area ratio)        227 non-null    object
 6   Net migration                       224 non-null    object
 7   Infant mortality (per 1000 births)  224 non-null    object
 8   GDP ($ per capita)                  226 non-null    float64
 9   Literacy (%)                        209 non-null    object
 10  Phones (per 1000)                   223 non-null    object
 11  Arable (%)                          225 non-null    object
 12  Crops (%)                           225 non-null    object
 13  Other (%)                           225 non-null    object
 14  Climate                             205 non-null    object
 15  Birthrate                           224 non-null    object
 16  Deathrate                           223 non-null    object
 17  Agriculture                         212 non-null    object
 18  Industry                            211 non-null    object
 19  Service                             212 non-null    object
dtypes: float64(1), int64(2), object(17)
memory usage: 35.6+ KB
```

```
[ ] columns_to_keep_as_int = ['Population', 'Area (sq. mi.)']
    columns_to_skip = ['Region', 'Country'] + columns_to_keep_as_int

    for col in df.columns:
        if col not in columns_to_skip and df[col].dtype == 'O':
            df[col] = df[col].str.replace(',', '').astype(float)
```

3 Data Analysis Part

```
# Total number of regions
total_regions = df['Region'].nunique()

# Total number of countries
total_countries = df['Country'].nunique()

# Total number of countries per each region
countries_per_region = df['Region'].value_counts()

print("Total number of regions:", total_regions)
print("Total number of countries:", total_countries)
print("Total number of countries per each region:")
print(countries_per_region)
```

```
Total number of regions: 11
Total number of countries: 227
Total number of countries per each region:
SUB-SAHARAN AFRICA                  51
LATIN AMER. & CARIB                 45
ASIA (EX. NEAR EAST)                28
WESTERN EUROPE                      28
OCEANIA                             21
NEAR EAST                           16
EASTERN EUROPE                      12
C.W. OF IND. STATES                 12
NORTHERN AFRICA                      6
NORTHERN AMERICA                     5
BALTICS                              3
Name: Region, dtype: int64
```

Average Regions GDP, Literacy, Agriculture

```
# Calculate the average for the specified columns by region
gdp_region_avg = round(df.groupby('Region')[['GDP ($ per capita)', 'Literacy (%)', 'Agriculture']].mean(),2)
gdp_region_avg
```

Region	GDP ($ per capita)	Literacy (%)	Agriculture
ASIA (EX. NEAR EAST)	8053.57	795.54	124.29
BALTICS	11300.00	997.33	21.00
C.W. OF IND. STATES	4000.00	987.25	192.00
EASTERN EUROPE	9808.33	974.67	83.17
LATIN AMER. & CARIB	8882.22	907.30	57.02
NEAR EAST	10456.25	799.56	48.06
NORTHERN AFRICA	5550.00	677.00	134.50
NORTHERN AMERICA	26100.00	977.00	5.20
OCEANIA	8247.62	900.10	105.98
SUB-SAHARAN AFRICA	2323.53	625.19	195.53

It shows the values of fbs and restecg with a bar graph and analyzing the restecg and fbs features.

Top 15 Countries GDP per capita

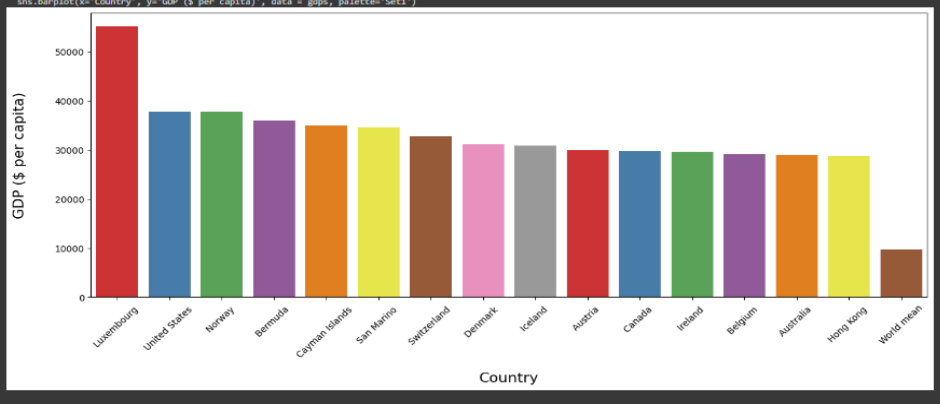
The resulting bar plot visually compares the GDP per capita of the top 15 countries with the global average, allowing for easy identification of the wealthiest nations.

```
fig, ax= plt.subplots(figsize=(16,6))
top_gdp_countries = df.sort_values('GDP ($ per capita)', ascending=False).head(15)
mean = pd.DataFrame({'Country':['World mean'], 'GDP ($ per capita)': [df['GDP ($ per capita)'].mean()]})
gdps = pd.concat([top_gdp_countries[['Country','GDP ($ per capita)']], mean],ignore_index=True)
sns.barplot(x='Country', y='GDP ($ per capita)', data = gdps, palette='Set1')
ax.set_xlabel(ax.get_xlabel(), labelpad=15)
ax.set_ylabel(ax.get_ylabel(), labelpad=30)
ax.xaxis.label.set_fontsize(16)
ax.yaxis.label.set_fontsize(16)
plt.xticks(rotation=45)
plt.show()
```

```
<ipython-input-19-c6401a0964a2>:5: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same effect.

  sns.barplot(x='Country', y='GDP ($ per capita)', data = gdps, palette='Set1')
```



Top Five Countries Literacy Rate and GDP for Asia Region

```
top_five_asia_countries_literacy = df[df['Region'].str.strip() == 'ASIA (EX. NEAR EAST)'].nlargest(5,'Literacy (%)')
top_five_asia_countries_literacy = top_five_asia_countries_literacy[['Country','Literacy (%)','GDP ($ per capita)']]
```

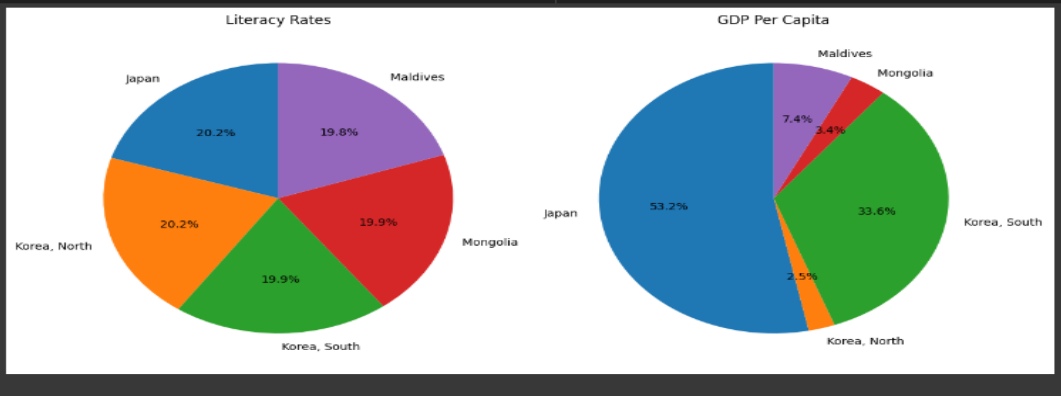
```
labels = top_five_asia_countries_literacy['Country']
literacy_rates = top_five_asia_countries_literacy['Literacy (%)']
gdp_values = top_five_asia_countries_literacy['GDP ($ per capita)']

fig,axes = plt.subplots(1,2,figsize=(12,6))


# Create a pie chart for literacy rates
axes[0].pie(literacy_rates, labels=labels, autopct='%1.1f%%', startangle=90)
axes[0].set_title('Literacy Rates')

# Create a pie chart for GDP per capita
axes[1].pie(gdp_values, labels=labels, autopct='%1.1f%%', startangle=90)
axes[1].set_title('GDP Per Capita')

plt.tight_layout()
plt.show()
```



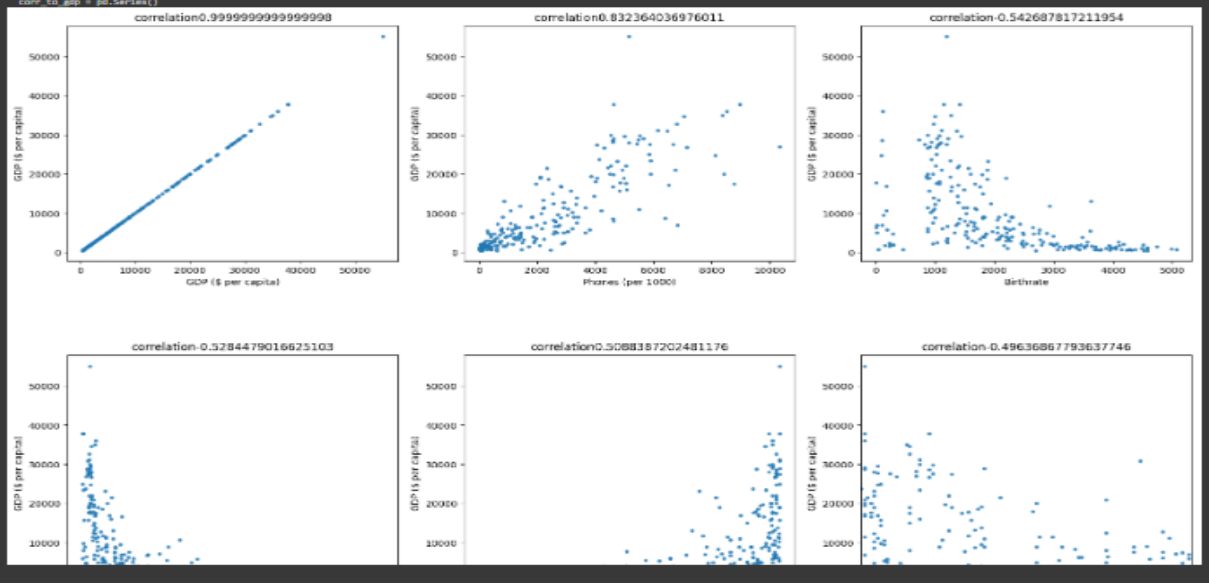
It shows the features of slope and ca form dataset values in bar graph format.

Correlation

```
fig, axes = plt.subplots(nrows=3, ncols=3, figsize=(20,20))
plt.subplots_adjust(hspace=0.4)
corr_to_gdp = pd.Series()
for col in df.columns.values[2:]:
    if ((col!="GDP ($ per capita)") & (col!="Climate")):
        corr_to_gdp[col] = df['GDP ($ per capita)'].corr(df[col])
abs_corr_to_gdp = corr_to_gdp.abs().sort_values(ascending=False)
corr_to_gdp = corr_to_gdp.loc[abs_corr_to_gdp.index]

for i in range(3):
    for j in range(3):
        sns.regplot(x=corr_to_gdp.index.values[i*3+j], y='GDP ($ per capita)', data=df, ax=axes[i,j],fit_reg=False, marker='.')
        title = "correlation"+str(corr_to_gdp[i*3+j])
        axes[i,j].set_title(title)
axes[1,2].set_xlim(0,102)
plt.show()
```

```
<ipython-input-24-1a90557749c1>:3: FutureWarning: The default dtype for empty Series will be 'object' instead of 'float64' in a future version. Specify a dtype explicitly to silence this warning.
  corr_to_gdp = pd.Series()
```



4. Modeling

Training and testing

First label encode the categorical features 'Regrion' and 'Climate', and while just use all features given in the dataset without further enginerring.

```
[ ] LE = LabelEncoder()
    df['Regional_label'] = LE.fit_transform(df['Region'])
```

```
[ ] X = df.drop(['Country','Region','GDP ($ per capita)'],axis=1)
    y = df['GDP ($ per capita)']
```

```
    X_train,X_test,y_train,y_test = train_test_split(X,y,test_size=0.3,shuffle=True)
```

```
[ ] X_train.shape, X_test.shape,y_train.shape,y_test.shape

    ((158, 18), (69, 18), (158,), (69,))
```

1 Linear Regresion

```
[ ] model_lr = LinearRegression()
    model_lr.fit(X_train,y_train)
    train_pred_y = model_lr.predict(X_train)
    test_pred_y = model_lr.predict(X_test)
    train_pred_y = pd.Series(train_pred_y.clip(0,train_pred_y.max()), index=y_train.index)
    test_pred_y = pd.Series(test_pred_y.clip(0,test_pred_y.max()), index=y_test.index)
```

Matrices

```
[ ] from sklearn.metrics import r2_score
    print("train score :", r2_score(train_pred_y,y_train))
    print('test score :',r2_score(test_pred_y,y_test))
    print("rmse_train :",np.sqrt(mean_squared_error(train_pred_y,y_train)))
    print("msle_train :", mean_squared_log_error(train_pred_y,y_train))
    print("rmse_test :", np.sqrt(mean_squared_error(test_pred_y,y_test)))
    print("msle_test :", mean_squared_log_error(test_pred_y,y_test))

    train score : 0.7122128373024303
    test score : 0.7908972994602097
    rmse_train : 5036.119822122485
    msle_train : 5.84330078352195
    rmse_test : 3636.4489893244277
    msle_test : 4.851101146991484
```

Finally, it shows that Random Forest algorithm has more accuracy than Logistic Regression from the dataset values and also shows the accuracy percentage of Random Forest Algorithm.

2 Random Forest

```
[ ] model_rf = RandomForestRegressor()
    model_rf.fit(X_train,y_train)
    train_pred_y = model_rf.predict(X_train)
    test_pred_y = model_rf.predict(X_test)
    train_pred_y = pd.Series(train_pred_y.clip(0,train_pred_y.max()), index=y_train.index)
    test_pred_y = pd.Series(test_pred_y.clip(0,test_pred_y.max()), index=y_test.index)
```

```
[ ] from sklearn.metrics import r2_score
    print("train score :", r2_score(train_pred_y,y_train))
    print('test score :',r2_score(test_pred_y,y_test))
    print("rmse_train :",np.sqrt(mean_squared_error(train_pred_y,y_train)))
    print("msle_train :", mean_squared_log_error(train_pred_y,y_train))
    print("rmse_test :", np.sqrt(mean_squared_error(test_pred_y,y_test)))
    print("msle_test :", mean_squared_log_error(test_pred_y,y_test))

    train score : 0.957884838949452
    test score : 0.8197820748362639
    rmse_train : 2135.7472960671703
    msle_train : 0.12505451063267148
    rmse_test : 3778.2863572129427
    msle_test : 0.1877503248262545
```

```
[ ] model_rf.predict(X_train.iloc[10].values.reshape(1,-1))

    /usr/local/lib/python3.10/dist-packages/sklearn/base.py:439: UserWarning: X does not have valid feature names, but RandomForestRegressor was fitted with feature names
      warnings.warn(
    array([1043.])
```

```
[ ] y_train.iloc[10]

    700.0
```

Chapter 6

SUMMARY AND CONCLUSION

6.1 Summary

This project objective is to predict the GDP Using Machine Learning. So, thispaper a machine learning algorithm is proposed for the implementation of a world Agdp prediction system which was validated on one open access world GDP prediction datasets.

6.2 Conclusion

In this project we have developed a predictor which is easy to understand our country or different country's GDP. It is important to know and to have the some basic information about the GDP. In this project viewer can easily understand the importance of features and how much they affected on the GDP, also the best and worst predictor performance using linear regressors

References

[1] Fernando J. Gross Domestic Product (GDP). Investopedia. 2021.

[2] Laine M. Introduction to dynamic linear models for time series analysis. InGeodetic Time Series Analysis in Earth Sciences 2020 (pp. 139-156). Springer, Cham.

[3] Lalitha VL, Raju SH, Sonti VK, Mohan VM. Customized Smart Object Detection: Statistics of detected objects using IoT. In2021 International Conference on Artificial Intelligence and Smart Systems (ICAIS) 2021 Mar 25 (pp. 1397-1405). IEEE.

[4] Qin T. Machine Learning Basics. InDual Learning 2020 (pp. 11-23). Springer, Singapore

[5] Brownlee J. A Gentle Introduction to the Gradient Boosting Algorithm for Machine Learning. 2016.

[6] Kurihara Y, Fukushima A. AR Model or Machine Learning for Forecasting GDP and Consumer Price for G7 Countries. Applied Economics and Finance. 2019;6(3):1-6.

[7] Richardson A, van Florenstein Mulder T, Vehbi T. Nowcasting GDP using machine-learning algorithms: A real-time assessment. International Journal of Forecasting. 2021 Apr 1;37(2):941-8.

[8] Cicceri G, Inserra G, Limosani M. A machine learning approach to forecast economic recessions—an italian case study. Mathematics. 2020 Feb;8(2):241.

9] Richardson A, Mulder T. Nowcasting New Zealand GDP using machine learning algorithms.

[10] Daga UR, Das R, Maheshwari B. Estimation, Analysis and Projection of India's GDP.

[11]Chen DH, Dahlman CJ. The knowledge economy, the KAM methodology and World Bank operations. World Bank Institute Working Paper. 2005 Oct 19(37256).

[12] Hawksworth J, Cookson G. The world in 2050. How big will the major emerging market economies get and how can the OECD compete. 2006 Se

APPENDIX:

STEPS FOR IMPLEMENTATION:

1. Install the required packages for building the _Passive Aggressive Classifiers‘.
2. Load the libraries into the workspace from the packages.
3. Read the input data set.
4. Normalize the given input dataset.
5. Divide this normalized data into two parts: A. Train data.

A. Test data (Note: 80% of Normalized data is used as Train data, 20%the Normalized data is used as Test data) **a. Sample code:**

#Importing essential libraries

import numpy as np import

pandas as pd import

matplotlib.pyplot as plt import

seaborn as sns %matplotlib

inline import os

print(os.listdir())

import warnings warnings.filterwarnings('ignore')

#Importing and understanding our dataset dataset

= pd.read_csv(—heart.csv")

#Verifying it as a 'dataframe' object in pandas type(dataset)

#Shape of dataset dataset.shape

```
#Printing out a few columns
dataset.head(5)
dataset.sample(5) #Description
the dataset dataset.describe()
#Describing dataset information dataset.info
()
#Let's understand our columns better info = ["age","1: male, 0: female","chest
pain type, 1: typical angina, 2: atypical angina, 3: non-anginal pain, 4:
asymptomatic","resting blood pressure"," serum cholestoral in mg/dl","fasting blood
sugar > 120 mg/dl","resting electrocardiographic results (values 0,1,2)"," maximum
heart rate achieved","exercise induced angina","oldpeak = ST depression induced
by exercise relative to rest","the slope of the peak exercise ST segment","number
of major vessels (0-3) colored by flourosopy","thal: 3 = normal; 6 = fixed defect; 7 =
reversable defect"] for i in range(len(info)):
   print(dataset.columns[i]+":\t\t\t"+info[i])
#Analysing the 'target' variable
dataset["target"].describe() dataset["target"].unique()
#Checking correlation between columns
print(dataset.corr()["target"].abs().sort_values(ascending=False)) #First,
analysing the target variable:
y = dataset["target"]
 sns.countplot(y) target_temp =
dataset.target.value_counts()
print(target_temp)
#printing the patience with or without heart problem print
("Percentage of patience without heart problems
"+str(round(target_temp[0]*100/303,2)))          print
("Percentage of patience with heart problems:
"+str(round(target_temp[1]*100/303,2)))
#We'll analyse 'sex', 'cp', 'fbs', 'restecg', 'exang', 'slope', 'ca' and 'thal' features
#Analysing the 'Sex' feature
dataset["sex"].unique() sns.barplot(dataset["sex"],y)
```

```
#Analysing the 'Chest Pain Type' feature
dataset["cp"].unique()
sns.barplot(dataset["cp"],y)  #Analysing
the FBS feature dataset["fbs"].describe()
dataset["fbs"].unique()
sns.barplot(dataset["fbs"],y)
#Analysing the restecg feature
dataset["restecg"].unique()
sns.barplot(dataset["restecg"],y)
#Analysing the 'exang' feature
dataset["exang"].unique()
sns.barplot(dataset["exang"],y)
#Analysing the Slope feature
dataset["slope"].unique()
sns.barplot(dataset["slope"],y) #Analysing
the 'ca' feature dataset["ca"].unique()
sns.countplot(dataset["ca"])
sns.barplot(dataset["ca"],y)
# Analysing the 'thal' feature
dataset["thal"].unique()
sns.barplot(dataset["thal"],y)
sns.distplot(dataset["thal"]) #Train Test split from
sklearn.model_selection import train_test_split
predictors = dataset.drop("target",axis=1) target =
dataset["target"] X_train,X_test,Y_train,Y_test =
train_test_split(predictors,target,test_size=0.20,random_state=0)
X_train.shape
X_test.shape
Y_train.shape
Y_test.shape #Model Fitting from
sklearn.metrics import accuracy_score
```

```
#Logistic Regression from sklearn.linear_model
import LogisticRegression lr = LogisticRegression()
lr.fit(X_train,Y_train) Y_pred_lr = lr.predict(X_test)
Y_pred_lr.shape
#Printing the accuracy score for Logistic Regression Algorithm score_lr =
round(accuracy_score(Y_pred_lr,Y_test)*100,2) print ("The accuracy score
achieved using Logistic Regression is: "+str(score_lr)+" %")
#Random Forest from sklearn.ensemble import
RandomForestClassifier max_accuracy = 0 for x in
range (2000):     rf =
RandomForestClassifier(random_state=x)
rf.fit(X_train,Y_train)
   Y_pred_rf = rf.predict(X_test)     current_accuracy =
round(accuracy_score(Y_pred_rf,Y_test)*100,2)
if(current_accuracy>max_accuracy):         max_accuracy =
current_accuracy         best_x = x
rf = RandomForestClassifier(random_state=best_x)
rf.fit(X_train,Y_train) Y_pred_rf = rf.predict(X_test)
Y_pred_rf.shape
#Printing the accuracy score for Random Forest Algorithm score_rf =
round(accuracy_score(Y_pred_rf,Y_test)*100,2)  print ("The accuracy score
achieved using Random Forest is: "+str(score_rf)+" %")
#Showing in bar graph format
sns.set(rc={'figure.figsize':(15,8)})
plt.xlabel("Algorithms") plt.ylabel("Accuracy
score") sns.barplot(algorithms,scores)
#Showing the accuracy of both algorithms scores = [score_lr,score_rf] algorithms
= ["Logistic Regression","Random Forest"]     for i in range(len(algorithms)):     print
("The accuracy score achieved using "+algorithms[i]+" is:‖+str(scores[I])+‖ %‖
```