

**VISVESVARAYA TECHNOLOGICAL UNIVERSITY,
BELAGAVI**



**A PROJECT REPORT ON
“STUDENTS COLLEGE SEAT ALLOTMENT
PREDICTION MODEL USING MACHINE LEARNING
TECHNIQUES”**

Submitted in partial fulfillment for the award of Degree of,

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE & ENGINEERING

By

DHANUSH V U

4AL15CS031

CHARANARAJ N

4AL16CS025

DERIL QUADRAS

4AL16CS031

JAYANTH S V

4AL16CS041

**Under the Guidance of
VASUDEV SHAHAPUR
Associate Professor**



**DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
ALVA'S INSTITUTE OF ENGINEERING AND TECHNOLOGY
MOODBIDRI-574225, KARNATAKA 2019 – 2020**

ALVA'S INSTITUTE OF ENGINEERING AND TECHNOLOGY
MIJAR, MOODBIDRI D.K. -574225, KARNATAKA



DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

CERTIFICATE

This is to certify that the project entitled **“STUDENTS COLLEGE SEAT ALLOTMENT PREDICTION MODEL USING MACHINE LEARNING TECHNIQUES”** has been successfully completed by

DHANUSH V U	4AL15CS031
CHARANARAJ N	4AL16CS025
DERIL QUADRAS	4AL16CS031
JAYANTH S V	4AL16CS041

the bonafide students of **DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING, ALVA'S INSTITUTE OF ENGINEERING AND TECHNOLOGY** of the **VISVESVARAYA TECHNOLOGICAL UNIVERSITY, BELAGAVI** during the year 2019–2020. It is certified that all corrections/suggestions indicated for Internal Assessment have been incorporated in the report deposited in the departmental library. The project report has been approved as it satisfies the academic requirements in respect of Project work prescribed for the Bachelor of Engineering Degree.

Vasudev Shahapur
Project Guide

Dr. Manjunath Kotari
Head of the department

Dr. Peter Fernandes
Principal

External Viva

Name of the Examiners

Signature with Date

- 1.
- 2.

ALVA'S INSTITUTE OF ENGINEERING AND TECHNOLOGY
MIJAR, MOODBIDRI D.K. -574225, KARNATAKA



DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

DECLARATION

We,

DHANUSH V U
CHARANARAJ N
DERIL QUADRAS
JAYANTH S V

hereby declare that the dissertation entitled **“STUDENTS COLLEGE SEAT ALLOTMENT PREDICTION MODEL USING MACHINE LEARNING TECHNIQUES”** is completed and written by us under the supervision of our guide **Dr. Manjunath Kotari, Professor and Head, Department of Computer and Engineering, Alvas Institute of Engineering and Technology, Moodbidri, in partial fulfillment of requirements for the award of the degree BACHELOR OF ENGINEERING in DEPARTMENT OF COMPUTER AND ENGINEERING of the VISVESVARAYA TECHNOLOGICAL UNIVERSITY, BELGAVI** during the academic year **2018 – 2019**. The dissertation report is original and it has not been submitted for any other degree in any university.

DHANUSH V U
CHARANARAJ N
DERIL QUDRAS
JAYANTH S V

4AL15CS031
4AL16CS025
4AL16CS031
4AL16CS041

ACKNOWLEDGEMENT

The satisfaction and euphoria that accompany a successful completion of any task would be incomplete without the mention of people who made it possible, success is the epitome of hard work and perseverance, but steadfast of all is encouraging guidance.

So, with gratitude we acknowledge all those whose guidance and encouragement served as beacon of light and crowned the effort with success.

We thank our project guide **Vasudev Shahapur**, Associate Professor in Department of Computer Science & Engineering, who has been our source of inspiration. He has been especially enthusiastic in giving his valuable guidance and critical reviews.

The selection of this project work as well as the timely completion is mainly due to the interest and persuasion of my project coordinator **Vasudev Shahapur**, Associate Professor, Department of Computer Science & Engineering. We will remember his contribution for ever.

We sincerely thank, **Dr. Manjunath Kotari**, Professor and Head, Department of Computer Science & Engineering who has been the constant driving force behind the completion of the project.

We thank Principal **Dr. Peter Fernandes**, for his constant help and support throughout.

We are also indebted to **Management of Alva's Institute of Engineering and Technology, Mijar, Moodbidri** for providing an environment which helped us in completing the project.

Also, we thank all the teaching and non-teaching staff of Department of Computer Science & Engineering for the help rendered.

Finally we would like to thank my parents and friends whose encouragement and support was invaluable.

DHANUSH V U	4AL15CS031
CHARANARAJ N	4AL16CS025
DERIL QUADRAS	4AL16CS031
JAYANTH S V	4AL16CS041

ABSTRACT

The ease of making better choices and making better decisions in terms of selecting colleges is the main aim of this system. Our analysis on colleges for the students makes easier for them to make accurate decision about their preferred colleges. For such analysis, it requires future possibilities from the past record data which can potentially make the predictions and recommendation for students. Our analysis with the machine learning classification methods would help giving probable accuracy and this requires analytical methods for predicting future recommendation. Today, most students make mistakes in their preference list due to lack of knowledge, improper and incorrect analysis of colleges and insecure predictions. Hence repent and regret after allotment. Our project will solve the general issue of the student community by using machine learning technology. In this system Random Forest and Decision Tree machine learning classification algorithm is going to use.

TABLE OF CONTENTS

CHAPTER NO.	DESCRIPTIONS NO.	PAGE NO
	ACKNOWLEDGEMENT.....	i
	ABSTRACT.....	ii
	LIST OF FIGURES.....	iii
	LIST OF SNAPSHOTS.....	iv
1.	INTRODUCTION	
	1.1 PROBLEM STATEMENT	1
	1.2 OBJECTIVES	1
2.	LITERATURE SURVEY	2-4
3.	SYSTEM REQUIREMENTS AND SPECIFICATION	
	3.1 PROPERTIES OF SRS	5
	3.2 PURPOSE OF REQUIREMENT DOCUMENT	5
	3.3 HARDWARE REQUIREMENTS	6
	3.4 SOFTWARE REQUIREMENTS	6
4.	SYSTEM ANALYSIS	
	4.1 EXISTING SYSTEM	8
	4.2 PROPOSED SYSTEM	8
5.	SYSTEM DESIGN	
	5.1 SYSTEM ARCHITECTURE	9
	5.2 FLOW OF CONTROL	9
	5.3 DATA FLOW DIAGRAM	11
	5.3 USE CASE DIAGRAM	13
6.	METHODOLOGY	14-50
7.	SYSTEM TESTING	
	7.1 TESTING OBJECTIVES	51

7.2 TEST TYPES	51
7.3 UNIT TESTING	52
7.4 INTEGRATION TESTING	52
7.5 SYSTEM TESTING	
7.5.1 VALIDATION TESTING	52
7.5.2 BLACK BOX TESTING	53
7.5.3 WHITE BOX TESTING	53
7.6 ACCEPTANCE TESTING	54
8. RESULTS	55-58
9. CONCLUSION	59
 REFERENCES.....	 60

LIST OF FIGURES

FIGURE NO.	DESCRIPTION	PAGE NO.
5.1	System Architecture	9
5.2	Flow Chart	10
5.3.1	Level 0 Data Flow Diagram	11
5.3.2	Level 1 Data Flow Diagram	12
5.3.3	Level 2 Data Flow Diagram	12
5.4	Use Case Diagram	13
6.1	Software Development Life Cycle	24
6.2	Steps Of Methodology	24
6.3	Confusion Matrix and formulae	27

LIST OF SNAPSHOTS

Figure No	Description	Page No
Snapshot 1	Load Dataset	55
Snapshot 2	Data Pre-processing	55
Snapshot 3	Aptitude test Data analysis	56
Snapshot 4	Olympiads Data analysis	56
Snapshot 5	Split data into Train and Test	57
Snapshot 6	Training Random Forest Model	57
Snapshot 7	Training Decision Tree Model	58
Snapshot 8	College Seat Allotment Prediction Model	58

CHAPTER 1

INTRODUCTION

At present, there are sixteen IITs in India, for which admission is governed by DTE (Directorate of Technical Education). DTE carries out the admission through CAP (Centralized Admission Process). The process is done through the cap rounds and is very confusing for student to analyse the perfect college. The student's needed to verify the documents at Facilitation Centre and are supposed to give their preference list of colleges. Then based on their Subject marks, Category, and other attributes, college is allotted to them in three or more consecutive forms. It's very difficult for the students to and out suitable colleges for them based on their Subject score, Aptitude Test, Technical Skills, English Skills, Olympiads, reading and writing skills, memory capability score, etc. Various colleges provide degree in IITs in various branches. Though analysis of colleges and their cut offs is required in order to get the most correct preference list. It is very tedious job for a student to understand about the suitable colleges which provides preferred branch and to analyses it's last years cut offs in order to predict whether that he can get one of those colleges in CAP.

1.1 PROBLEM STATEMENT

- Most of the students make mistakes in their preference list due to lack of knowledge, improper and incorrect analysis of colleges and insecure predictions.
- Hence those students regret after what they get the college after allotment.

1.2 OBJECTIVES

- The main objective of this project is to predict the College and Department allocation for the students based on their marks and skills.

CHAPTER 2

LITERATURE REVIEW

[1]. Kandapriya Basu, Treena Basu, Ron Buckmire and Nishu Lal “**Predictive Models of Student College Commitment Decisions Using Machine Learning**”, Every year, academic institutions invest considerable effort and substantial resources to influence, predict and understand the decision-making choices of applicants who have been offered admission. In this study, we applied several supervised machine learning techniques to four years of data on 11,001 students, each with 35 associated features, admitted to a small liberal arts college in California to predict student college commitment decisions. By treating the question of whether a student offered admission will accept it as a binary classification problem, we implemented a number of different classifiers and then evaluated the performance of these algorithms using the metrics of accuracy, precision, recall, *F*-measure and area under the receiver operator curve. The results from this study indicate that the logistic regression classifier performed best in modelling the student college commitment decision problem, i.e., predicting whether a student will accept an admission offer, with an AUC score of 79.6%. The significance of this research is that it demonstrates that many institutions could use machine learning algorithms to improve the accuracy of their estimates of entering class sizes, thus allowing more optimal allocation of resources and better control over net tuition revenue.

[2]. Annam Mallikharjuna Roa, Nagineni Dharani, A. Satya Raghava, J. Buvanambigai, and K. Sathish Says in the paper “**College Admission Predictor**” System is a web based application system in which students can register their marks along with their personal information. This helps to predict their admissions in colleges. Administrator can add the college details and the batch details. Using this Application, the entrance seat allotment becomes easier and efficient. The main advantage of the project is the computerization of the entrance seat allotment process. Administrator has the power for the allotment. Admin can add the allotted seats into a file and the details are saved into the system. The total time for the entrance allotment becomes lower and the allotment process becomes faster. It helps students to make right decisions for choosing their college. In which students can register with their personal as well as marks details to prediction the admission in colleges and the administrator can allot the seats for the students. Administrator can add the college details and the batch details. Using this Application, the entrance seat allotment became easier and can be implemented using system. The main advantage of

the project is the computerization of the entrance seat allotment process. Administrator has the power for the allotment. Admin can add the allotted seats into a file and the details are saved into the system. The total time for the entrance allotment became lesser and the allotment process became faster. It helps student for making decision for choosing a right college

[3]. Charushila Patil, Akshay Diwate, Jayesh Baviskar, and Tejas Gholap Says in the paper **“EFFICIENT CAP ROUND PREDICTION FOR STUDENTS”**. In the earlier time, the students who get passed out from diploma where having problem for the admission of colleges for higher studies. It is observed that the students waste most of the time by checking on web sites like DTE (Director of Technical Education) in this website, the students have to search and download a specific PDF to get the particular information of college or cap round list. We can say that this website is slightly helpful to the students but it is not user-friendly .The GUI of DTE’s main drawback is any random student cannot access this websites because of its GUI .By looking to this problem ,we developed an new web portal for cap round admission process .In our web portal ,we gave a very user-friendly GUI to it so that any student can easily access It .The cap predictor system helps the diploma passes out students by reducing their efforts for searching on other site for hours. The system helps student by fetching the data of colleges. On the basis of information provided by the students .It gives accurate prediction of the college.

[4]. Anthony Dalton, Justin Beer, Sriharshasai Kommanapalli, and James S. Lanich, **“Machine Learning to Predict College Course Success”** purpose of paper was to present an analysis of the predictive ability of machine learning on students’ transfer-level course success in the California Community College system. The California Legislature passed assembly bill 705 in order to better place students in non-remedial coursework based on high school transcripts in order to increase college completion. Through utilizing machine learning methods on de-identified student high school transcript data, we create a predictive model for placing students in the correct initial college courses. The prediction is whether or not a student will be successful in transfer-level English and Mathematics courses. Industry knowledge is our guide to selecting variables to utilize in machine learning. The English dataset’s best prediction model is the Random Forest with an accuracy of 65.7% along with better precision, recall, and support values than the other models. The Mathematics dataset’s best prediction model is Logistic Regression at an accuracy of 67.1% and better precision, recall, and support values. The

results indicate that students' subject cumulative grade point average is the best predictor for the English model and without subject cumulative grade point average is the best predictor for the Mathematics model. Overall, we built a predictive model that can be replicated and uniformly implemented which provides up to two times better accuracy as compared to current admission standards. Implementation of our model will benefit first time college students by accurate placement into courses they will be able to successfully complete.

[5]. Rahul Sathawane, Rohan Battulwar, Prasheel Fuley, Ananiya Mahajan, Shivam Joshi⁵, and Roshan Chaturpale **“College Guesstimate”** This paper describes the analysis on colleges for the students makes easier for them to make accurate decision about their preferred colleges. For such analysis, it requires future possibilities from the past record data from DTE which can potentially make the predictions and recommendation for students. Our analysis with the data mining methods would help giving probable accuracy and this requires analytical methods for predicting future recommendation. Today, most students make mistakes in their preference list due to lack of knowledge, improper and incorrect analysis of colleges and insecure predictions. Hence repent and regret after allotment. Our project will solve the general issue of the student community by using technology.

CHAPTER 3

SYSTEM REQUIREMENTS AND SPECIFICATION

A Software Requirements Specification (SRS) is a complete description of the behaviour of the system to be developed. SRS is a document that completely describes what the proposed software should do without describing how the software will do it. It is two-way policy that both the client and the organization understand the requirements at any given point of time.

SRS document itself is precise and it provides the functions and capabilities of a system that it should provide. The basic purpose of SRS is to bridge the communication gap between the parties involved in the development of the software. It serves as an input to design specification. It also serves as the parent document to subsequent documents. Therefore, the SRS should be easy to understand and also should contain sufficient details in the system requirements so that a design solution can be devised easily.

3.1 PROPERTIES OF SRS

A good SRS is:

- **Correct:** An SRS is complete if the responses of the software to all classes of input data are specified in the SRS.
- **Complete:** An SRS is unambiguous if and only if every requirement stated has one and only one interpretation.
- **Unambiguous:** An SRS is verifiable if and only if the stated requirement is verifiable.
- **Verifiable:** A requirement is verifiable if there is some cost effective process that can check whether final software meets the requirement.
- **Modifiable:** An SRS is modifiable if its structure and style are such that any necessary changes can be made easily preserving completeness and consistency.

3.2 PURPOSE OF REQUIREMENT DOCUMENT

The document gives the detailed description of the both functional and non-functional Requirements. The purpose of the product requirements document (PRD) or product specifications to clearly and unambiguously articulate the product's purpose, features, functionality, and behaviour

3.3 HARDWARE REQUIREMENTS

System Processor : Core i5 8th Gen.

Hard Disk : 500GB.

RAM : 4GB .

3.4 SOFTWARE REQUIREMENTS

Operating System : Windows 10.

Programming Language : Python.

Framework : Anaconda.

IDE : Jupyter Notebook

CHAPTER 4

SYSTEM ANALYSIS

It is a process of collecting and interpreting facts, identifying the problems, and decomposition of a system into its components. System analysis is conducted for the purpose of studying a system or its parts in order to identify its objectives. It is a problem solving technique that improves the system and ensures that all the components of the system work efficiently to accomplish their purpose. Analysis specifies what the system should do.

A system must have three basic constraints:

1. A system must have some structure and behavior which is designed to achieve a predefined objective.
2. Interconnectivity and interdependence must exist among the system components.
3. The objectives of the organization have a higher priority than the objectives of its subsystems.

A system has the following properties:

Organization: Organization implies structure and order. It is the arrangement of components that helps to achieve predetermined objectives.

Interaction: It is defined by the manner in which the components operate with each other. For example, in an organization, purchasing department must interact with production department and payroll with personnel department.

Interdependence: Interdependence means how the components of a system depend on one another. For proper functioning, the components are coordinated and linked together according to a specified plan. The output of one subsystem is the required by other subsystem as input.

Integration: Integration is concerned with how a system components are connected together. It means that the parts of the system work together within the system even if each part performs a unique function.

Central organization: The objective of system must be central. It may be real or stated. It is not uncommon for an organization to state an objective and operate to achieve another. The users must know the main objective of a computer application early in the analysis for a successful design and conversion.

4.1 EXISTING SYSTEM

Making a wise career decision is very important for everyone. In recent years, decision support tools and mechanisms have assisted us in making the right career decisions. This enable a student who wishes to pursue Engineering, make up good decisions, using the help of a Decision Support System. Last 3 years' information has been obtained from the website of Directorate of Technical Education, India (DTE) which makes it freely available. Using Decision Rules, results are computed from which a student can choose which stream and college he/she can opt for on the basis of Entrance Exam marks he/she has scored. To make the results more relevant, a search in the already created decision system is performed. A student has to enter his/her Entrance Exam scores and the stream he/she wishes to opt for. Based on the entered information, the decision system will return colleges and streams categorized as Ambitious, Best Bargain and Safe.

4.1 PROPOSED SYSTEM

The system consists of distinct modules like Data Acquisition and Preprocessing, Feature Selection and Data Preparation, Model Construction and Model Training, and Model Validation and Result Analysis. In addition to these there are making better choices and making better decisions in terms of selecting colleges is the main aim of this system. Our analysis on colleges for the students makes easier for them to make accurate decision about their preferred colleges. For such analysis, it requires future possibilities from the past record data which can potentially make the predictions and recommendation for students. Our analysis with the machine learning classification methods would help giving probable accuracy and this requires analytical methods for predicting future recommendation. Today, most students make mistakes in their preference list due to lack of knowledge, improper and incorrect analysis of colleges and insecure predictions. Hence repent and regret after allotment. Our project will solve the general issue of the student community by using machine learning technology. In this system using Random Forest and Decision Tree machine learning classification algorithm.

CHAPTER 5

SYSTEM DESIGN

5.1 SYSTEM ARCHITECTURE

System Architecture design-identifies the overall hypermedia structure for the WebApp. Architecture design is tied to the goals establish for a WebApp, the content to be presented, the users who will visit, and the navigation philosophy that has been established. Content architecture, focuses on the manner in which content objects and structured for presentation and navigation. WebApp architecture, addresses the manner in which the application is structure to manage user interaction, handle internal processing tasks, effect navigation, and present content. WebApp architecture is defined within the context of the development environment in which the application is to be implemented.

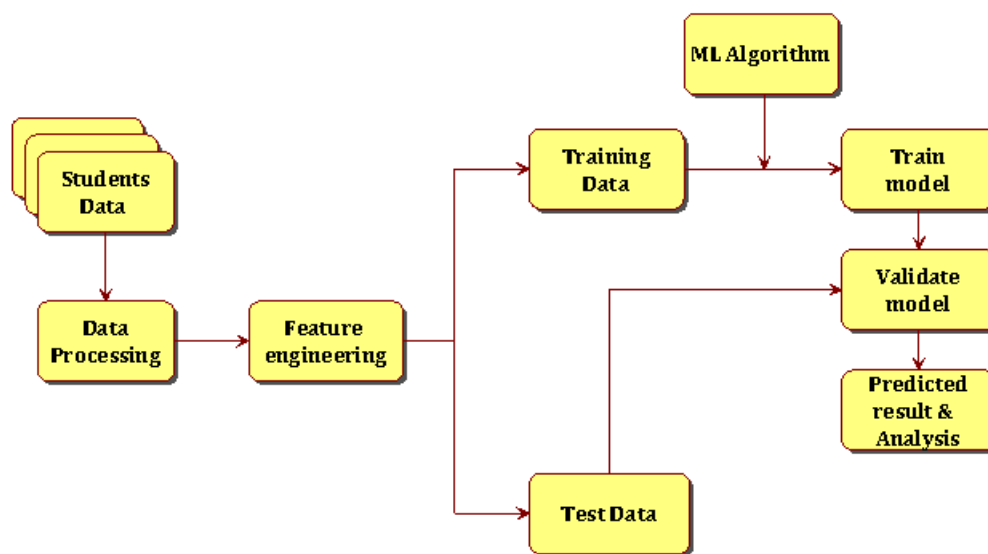


Fig 5.1 : System Architecture

5.2 FLOW OF CONTROL

It is important to complete all tasks and meet deadlines. There are many project management tools that are available to help project managers manage their tasks and schedule and one of them is the flowchart.

A flowchart is one of the seven basic quality tools used in project management and it displays the actions that are necessary to meet the goals of a particular task in the most practical sequence. Also called as process maps, this type of tool displays a series of

steps with branching possibilities that depict one or more inputs and transforms them to outputs.

The advantage of flowcharts is that they show the activities involved in a project including the decision points, parallel paths, branching loops as well as the overall sequence of processing through mapping the operational details within the horizontal value chain. Moreover, this particular tool is very used in estimating and understanding the cost of quality for a particular process. This is done by using the branching logic of the workflow and estimating the expected monetary returns.

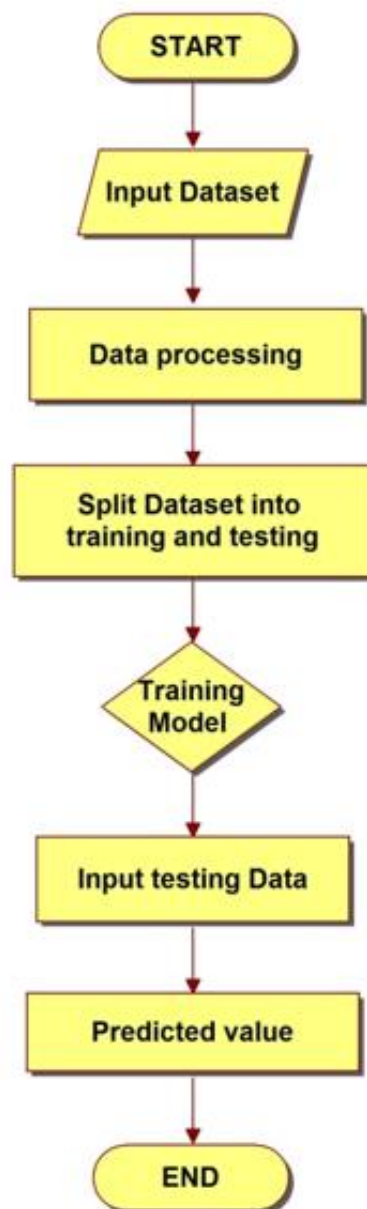


Fig 5.2 : Flow Chart Diagram

5.3 DATA FLOW DIAGRAM

A data flow diagram (DFD) is graphic representation of the "flow" of data through an information system. A data flow diagram can also be used for the visualization of data processing (structured design). It is common practice for a designer to draw a context level DFD first which shows the interaction between the system and outside entities. DFD's show the flow of data from external entities into the system, how the data moves from one process to another, as well as its logical storage.

There are only four symbols:

1. Squares representing external entities, which are sources and destinations of information entering and leaving the system.
2. Rounded rectangles representing processes, in other methodologies, may be called 'Activities', 'Actions', 'Procedures', 'Subsystems' etc. which take data as input, do processing to it, and output it.
3. Arrows representing the data flows, which can either, be electronic data or physical items. It is impossible for data to flow from data store to data store except via a process, and external entities are not allowed to access data stores directly.
4. The flat three-sided rectangle is representing data stores should both receive information for storing and provide it for further processing.

5.3.1 LEVEL 0 DATA FLOW DIAGRAM

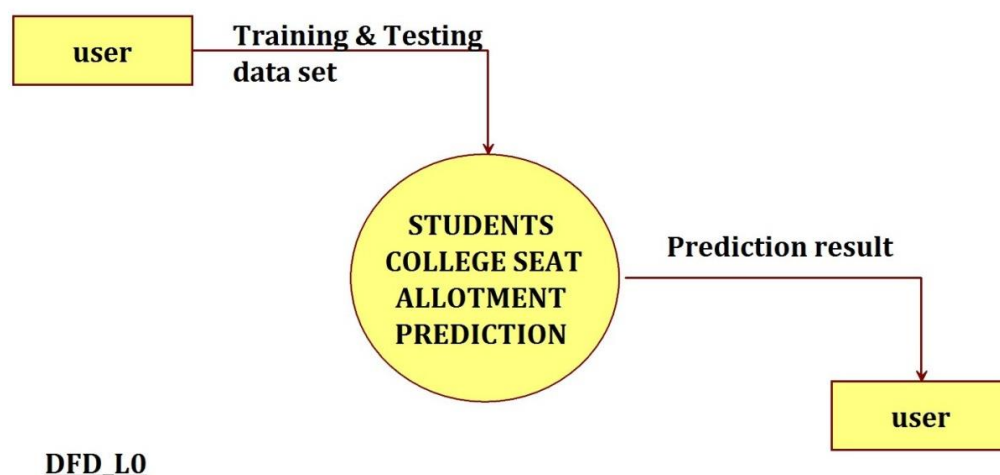


Fig 5.3.1 : Level 0 Data flow Diagram

5.3.2 LEVEL 1 DATA FLOW DIAGRAM

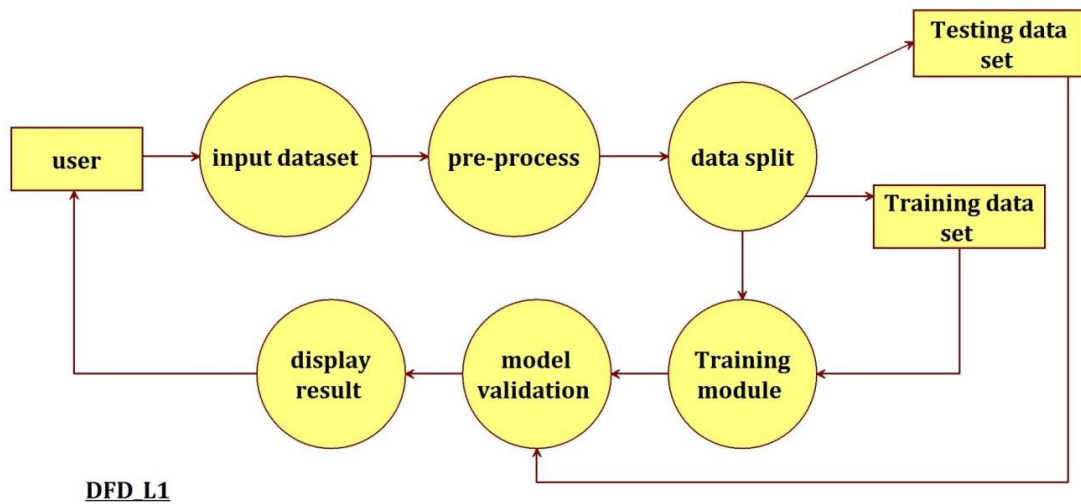


Fig 5.3.2 : Level 1 Data Flow Diagram

5.3.3 LEVEL 2 DATA FLOW DIAGRAM

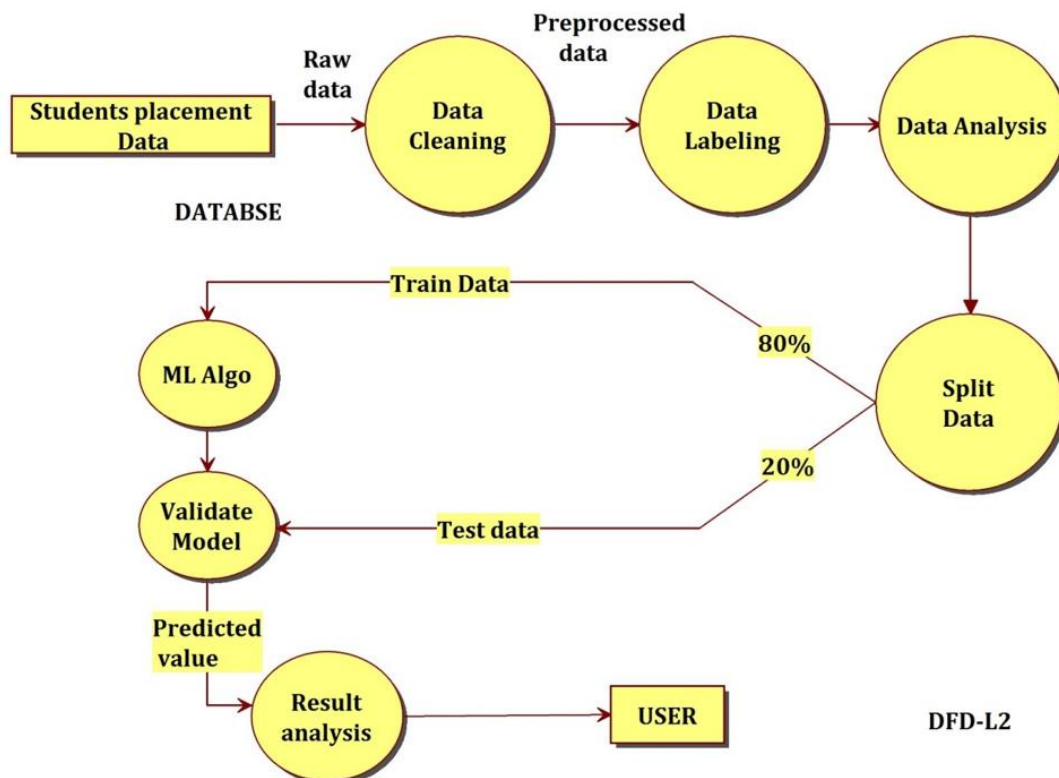


Fig 5.3.3 : Level 2 Data Flow Diagram

5.4 USE CASE DIAGRAM

A use case is a set of scenarios that describing an interaction between a source and a destination. A use case diagram displays the relationship among actors and use cases. The two main components of a use case diagram are use cases and actors. shows the use case diagram.

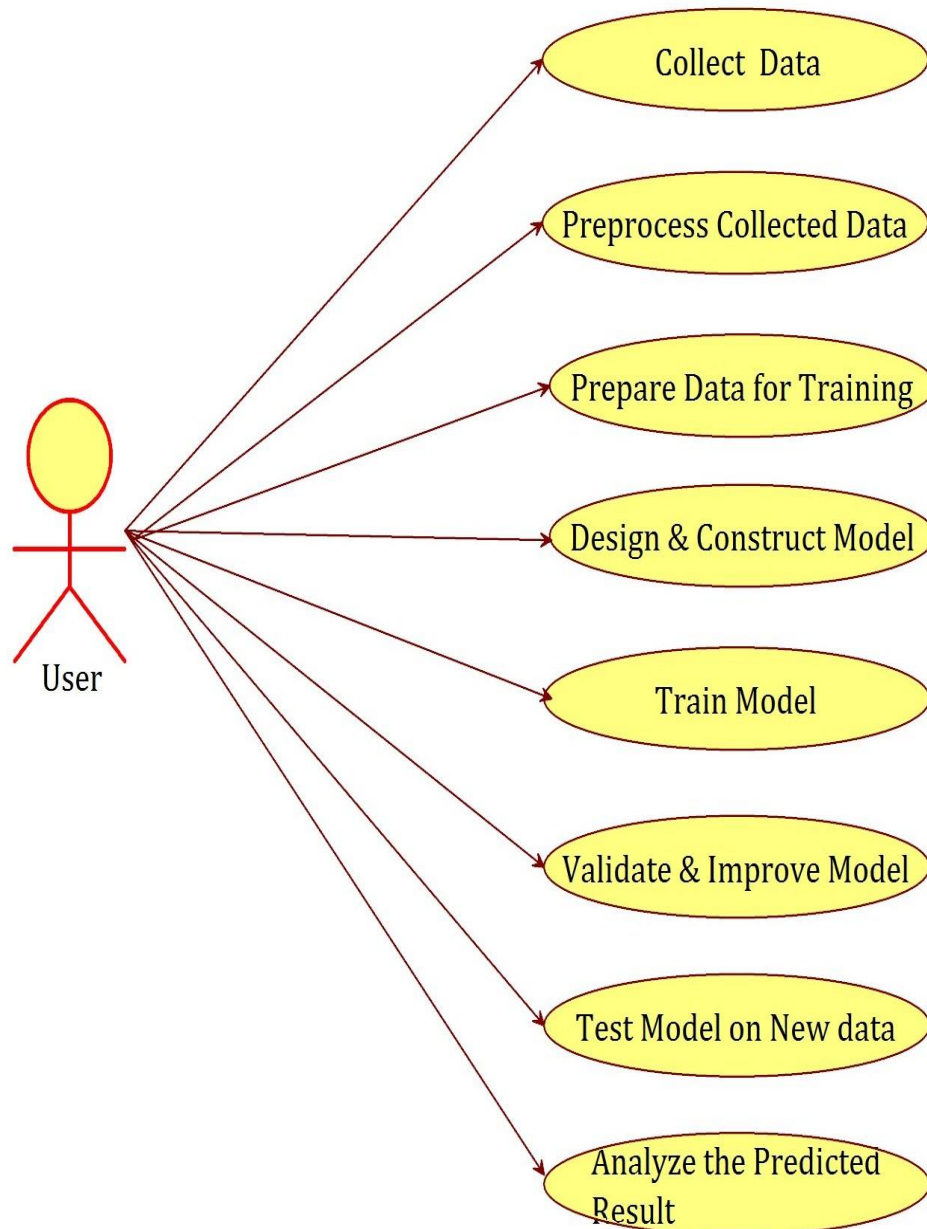


Fig 5.4 :Use Case Diagram

CHAPTER 6

METHODOLOGY

Python is an easy to learn, powerful programming language. It has efficient high-level data structures and a simple but effective approach to object-oriented programming. Python's elegant syntax and dynamic typing, together with its interpreted nature, make it an ideal language for scripting and rapid application development in many areas on most platforms.

The Python interpreter and the extensive standard library are freely available in source or binary form for all major platforms from the Python Web site, <https://www.python.org/>, and may be freely distributed. The same site also contains distributions of and pointers to many free third party Python modules, programs and tools, and additional documentation.

The Python interpreter is easily extended with new functions and data types implemented in C or C++ (or other languages callable from C). Python is also suitable as an extension language for customizable applications.

Python is a high-level, interpreted, interactive and object-oriented scripting language. Python is designed to be highly readable. It uses English keywords frequently where as other languages use punctuation, and it has fewer syntactical constructions than other languages.

- **Python is Interpreted** – Python is processed at runtime by the interpreter. You do not need to compile your program before executing it. This is similar to PERL and PHP.
- **Python is Interactive** – you can actually sit at a Python prompt and interact with the interpreter directly to write your programs.
- **Python is Object-Oriented** – Python supports Object-Oriented style or technique of programming that encapsulates code within objects.
- **Python is a Beginner's Language** – Python is a great language for the beginner-level programmers and supports the development of a wide range of applications from simple text processing to WWW browsers to games.

Features:

Python's features include – All these.

- **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-maintain.
- **A broad standard library** – Python's bulk of the library is very portable and cross-platform compatible on UNIX, Windows, and Macintosh.
- **Interactive Mode** – Python has support for an interactive mode which allows interactive testing and debugging of snippets of code.
- **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.
- **Scalable** – Python provides a better structure and support for large programs than shell scripting.

Why python emerging as a leader:

There's battle out there happening in the minds of aspiring data scientists to choose the best data science tool. Though there are quite a number of data science tools that provide the much-needed option, the close combat narrows down between two popular languages – Python and R.

Between the two, Python is emerging as the popular language used more in data science applications. Take the case of the tech giant Google that has created the deep

learning framework called tensor flow – Python is the primary language used for creating this framework. Its footprint has continued to increase in the environment promoted by Netflix. Production engineers at Face book and Khan Academy have for long been using it as a prominent language in their environment.

Python has other advantages that speed up it's upward swing to the top of data science tools. It integrates well with the most cloud as well as platform-as-a-service providers. In supporting multiprocessing for parallel computing, it brings the distinct advantage of ensuring large-scale performance in data science and machine learning. Python can also be extended with modules written in C/C++.

Where Python becomes the perfect-fit:

There are tailor-made situations where it is the best data science tool for the job. It is perfect when data analysis tasks involve integration with web apps or when there is a need to incorporate statistical code into the production database. The full-fledged programming nature of Python makes it a perfect fit for implementing algorithms. Its packages rooted for specific data science jobs. Packages like Numpy, SciPy, and pandas produce good results for data analysis jobs. While there is a need for graphics, Python's matplotlib emerges as a good package, and for machine learning tasks, scikit-learn becomes the ideal alternate.

Why is Python preferred over other data science tools?

It is 'Pythonic' when the code is written in a fluent and natural style. Apart from that, it is also known for other features that have captured the imaginations of data science community.

Easy to learn:

The most alluring factor of Python is that anyone aspiring to learn this language can learn it easily and quickly. When compared to other data science languages like R, Python promotes a shorter learning curve and scores over others by promoting an easy-to-understand syntax.

Scalability:

When compared to other languages like R, Python has established a lead by emerging as a scalable language, and it is faster than other languages like Matlab and Stata. Python's

scalability lies in the flexibility that it gives to solve problems, as in the case of YouTube that migrated to Python. Python has come good for different usages in different industries and for rapid development of applications of all kinds.

Choice of data science libraries:

The significant factor giving the push for Python is the variety of data science/data analytics libraries made available for the aspirants. Pandas, StatsModels, NumPy, SciPy, and Scikit-Learn, are some of the libraries well known in the data science community. Python does not stop with that as libraries have been growing over time. What you thought was a constraint a year ago would be addressed well by Python with a robust solution addressing problems of specific nature.

Python community:

One of the reasons for the phenomenal rise of Python is attributed to its ecosystem. As Python extends its reach to the data science community, more and more volunteers are creating data science libraries. This, in turn, has led the way for creating the most modern tools and processing in Python.

The widespread and involved community promotes easy access for aspirants who want to find solutions to their coding problems. Whatever queries you need, it is a click or a Google search away. Enthusiasts can also find access to professionals on Code mentor and Stack Overflow to find the right answers for their queries.

Graphics and visualization:

Python comes with varied visualization options. Matplotlib provides the solid foundation around which other libraries like Sea born, pandas plotting, and ggplot have been built. The visualization packages help you get a good sense of data, create charts, graphical plot and create web-ready interactive plots.

Why Choose Python?

If you're going to write programs, there are literally dozens of commonly used languages to choose from. Why choose Python? Here are some of the features that make Python an appealing choice.

Python is Popular

Python has been growing in popularity over the last few years. The 2018 Stack Overflow Developer Survey ranked Python as the 7th most popular and the number one most wanted technology of the year. World-class software development countries around the globe use Python every single day.

According to research by Dice Python is also one of the hottest skills to have and the most popular programming language in the world based on the. Popularity of Programming Language Index

Due to the popularity and widespread use of Python as a programming language, Python developers are sought after and paid well. If you'd like to dig deeper into Python salary statistics and job opportunities, you can do so here.

Many languages are compiled, meaning the source code you create needs to be translated into machine code, the language of your computer's processor, before it can be run. Programs written in an interpreted language are passed straight to an interpreter that runs them directly.

This makes for a quicker development cycle because you just type in your code and run it, without the intermediate compilation step.

One potential downside to interpreted languages is execution speed. Programs that are compiled into the native language of the computer processor tend to run more quickly than interpreted programs. For some applications that are particularly computationally intensive, like graphics processing or intense number crunching, this can be limiting.

In practice, however, for most programs, the difference in execution speed is measured in milliseconds, or seconds at most, and not appreciably noticeable to a human user. The expediency of coding in an interpreted language is typically worth it for most applications.

Python is Free

The Python interpreter is developed under an OSI-approved open-source license, making it free to install, use, and distribute, even for commercial purposes. A version of the interpreter is available for virtually any platform there is, including all flavors of Unix,

Windows, macOS, smartphones and tablets, and probably anything else you ever heard of. A version even exists for the half dozen people remaining who use OS/2.

Python is Portable

Because Python code is interpreted and not compiled into native machine instructions, code written for one platform will work on any other platform that has the Python interpreter installed. (This is true of any interpreted language, not just Python.)

Python is Simple

As programming languages go, Python is relatively uncluttered, and the developers have deliberately kept it that way. A rough estimate of the complexity of a language can be gleaned from the number of keywords or reserved words in the language. These are words that are reserved for special meaning by the compiler or interpreter because they designate specific built-in functionality of the language. Python 3 has 33 keywords, and Python 2 has 31. By contrast, C++ has 62, Java has 53, and Visual Basic has more than 120, though these latter examples probably vary somewhat by implementation or dialect. Python code has a simple and clean structure that is easy to learn and easy to read. In fact, as you will see, the language definition enforces code structure that is easy to read.

But It's Not That Simple

For all its syntactical simplicity, Python supports most constructs that would be expected in a very high-level language, including complex dynamic data types, structured and functional programming, and object-oriented programming.

Additionally, a very extensive library of classes and functions is available that provides capability well beyond what is built into the language, such as database manipulation or GUI programming.

Python accomplishes what many programming languages don't: the language itself is simply designed, but it is very versatile in terms of what you can accomplish with it.

Is Python 'the' tool for machine learning?

When it comes to data science, machine learning is one of the significant elements used to maximize value from data. With Python as the data science tool, exploring the basics of machine learning becomes easy and effective. In a nutshell, machine learning is more

about statistics, mathematical optimization, and probability. It has become the most preferred machine learning tool in the way it allows aspirants to ‘do math’ easily.

Name any math function, and you have a Python package meeting the requirement. There is Numpy for numerical linear algebra, CVXOPT for convex optimization, Scipy for general scientific computing, SymPy for symbolic algebra, PYMC3, and Statsmodel for statistical modeling.

With the grip on the basics of machine learning algorithm including logistic regression and linear regression, it makes it easy to implement machine learning systems for predictions by way of its scikit-learn library. It’s easy to customize for neural networks and deep learning with libraries including Keras, Theano, and TensorFlow.

Data science landscape is changing rapidly, and tools used for extracting value from data science have also grown in numbers. The two most popular languages that fight for the top spot are R and Python. Both are revered by enthusiasts, and both come with their strengths and weaknesses. But with the tech giants like Google showing the way to use Python and with the learning curve made short and easy, it inches ahead to become the most popular language in the data science world.

Machine learning

Machine Learning is a method of statistical learning where each instance in a dataset is described by a set of features or attributes. In contrast, the term “Deep Learning” is a method of statistical learning that extracts features or attributes from raw data. Deep Learning does this by utilizing neural networks with many hidden layers, big data, and powerful computational resources. The terms seem somewhat interchangeable, however, with Deep Learning method, The algorithm constructs representations of the data automatically. In contrast, data representations are hard-coded as a set of features in machine learning algorithms, requiring further processes such as feature selection and extraction, (such as PCA).

Both of these terms are in dramatic contrast with another class of classical artificial intelligence algorithms known as Rule-Based Systems where each decision is manually programmed in such a way that it resembles a statistical model.

In Machine Learning and Deep Learning, there are many different models that fall into two different categories, supervised and unsupervised. In unsupervised learning, algorithms such as k-Means, hierarchical clustering, and Gaussian mixture models

attempt to learn meaningful structures in the data. Supervised learning involves an output label associated with each instance in the dataset. This output can be discrete/categorical or real-valued. Regression models estimate real-valued outputs, whereas classification models estimate discrete-valued outputs. Simple binary classification models have just two output labels, 1 (positive) and 0 (negative). Some popular supervised learning algorithms that are considered Machine Learning: are linear regression, logistic regression, decision trees, support vector machines, and neural networks, as well as non-parametric models such as k-Nearest Neighbors.

Supervised Machine Learning

The majority of practical machine learning uses supervised learning.

Supervised learning is where you have input variables (x) and an output variable (Y) and you use an algorithm to learn the mapping function from the input to the output.

$$Y = f(X)$$

The goal is to approximate the mapping function so well that when you have new input data (x) that you can predict the output variables (Y) for that data.

It is called supervised learning because the process of an algorithm learning from the training dataset can be thought of as a teacher supervising the learning process. We know the correct answers, the algorithm iteratively makes predictions on the training data and is corrected by the teacher. Learning stops when the algorithm achieves an acceptable level of performance.

Supervised learning problems can be further grouped into regression and classification problems.

- **Classification:** A classification problem is when the output variable is a category, such as “red” or “blue” or “disease” and “no disease”.
- **Regression:** A regression problem is when the output variable is a real value, such as “dollars” or “weight”.

Some common types of problems built on top of classification and regression include recommendation and time series prediction respectively.

Some popular examples of supervised machine learning algorithms are:

- Linear regression for regression problems.
- Random forest for classification and regression problems.
- Support vector machines for classification problems.

Unsupervised Machine Learning

Unsupervised learning is where you only have input data (X) and no corresponding output variables.

The goal for unsupervised learning is to model the underlying structure or distribution in the data in order to learn more about the data.

These are called unsupervised learning because unlike supervised learning above there is no correct answers and there is no teacher. Algorithms are left to their own devices to discover and present the interesting structure in the data.

Unsupervised learning problems can be further grouped into clustering and association problems.

- **Clustering:** A clustering problem is where you want to discover the inherent groupings in the data, such as grouping customers by purchasing behavior.
- **Association:** An association rule learning problem is where you want to discover rules that describe large portions of your data, such as people that buy X also tend to buy Y.

Some popular examples of unsupervised learning algorithms are:

- k-means for clustering problems.
- Apriori algorithm for association rule learning problems.

Semi-Supervised Machine Learning

Problems where you have a large amount of input data (X) and only some of the data is labeled (Y) are called semi-supervised learning problems.

These problems sit in between both supervised and unsupervised learning.

A good example is a photo archive where only some of the images are labeled, (e.g. dog, cat, person) and the majority are unlabeled to domain experts. Whereas unlabeled data is cheap and easy to collect and store.

You can use unsupervised learning techniques to discover and learn the structure in the input variables.

You can also use supervised learning techniques to make best guess predictions for the unlabeled data, feed that data back into the supervised learning algorithm as training data and use the model to make predictions on new unseen data.

Data Size:

Both Machine Learning and Deep Learning are able to handle massive dataset sizes, however, machine learning methods make much more sense with small datasets. For example, if you only have 100 data points, decision trees, k-nearest neighbors, and other machine learning models will be much more valuable to you than fitting a deep neural network on the data. This is due to the next topic of difference, Interpretability

Interpretability:

Example for how interpretability works in ML & DL:



A lot of the criticism of deep learning methods and machine learning algorithms such as Support Vector Machine or (maybe, because you can at least visualize the constituent probabilities making up the output), Naive Bayes, are due to their difficulty to interpret. For example, when a Convolutional Neural Network outputs 'cat' in a dog vs. cat problem, nobody seems to know why it did that. In contrast, when you are modeling data such as an electronic health record or bank loan dataset with a machine learning technique, it is much easier to understand the reasoning for the model's prediction.

One of the best examples of interpretability is decision trees where you follow logical tests down nodes of the tree until you reach a decision. Another machine learning algorithm with high interpretability is k-Nearest Neighbors. This is not a parametric learning algorithm but still falls under the category of machine learning algorithms. It is very interpretability because you easily reason about the similar instances for yourself.

SDLC

It will cover the details explanation of methodology that is being used to make this project complete and working well. Many methodology or findings from this field mainly generated into journal for others to take advantages and improve as upcoming studies. The method is use to achieve the objective of the project that will accomplish a perfect result. In order to evaluate this project, the methodology based on System Development Life Cycle (SDLC), generally three major step, which is planning, implementing and analysis.

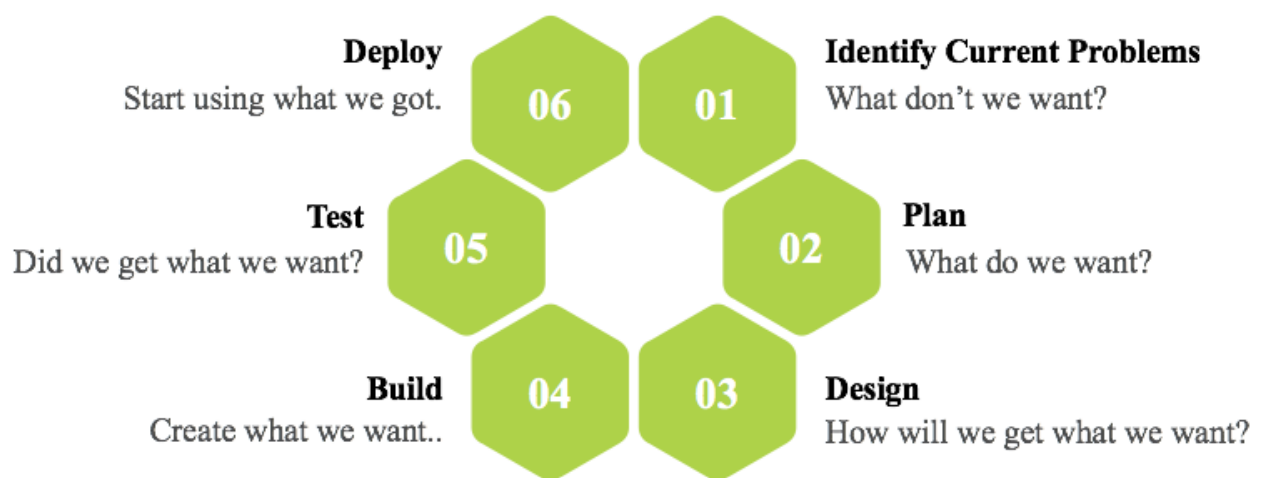


Fig 6.1 : Software Development Life Cycle

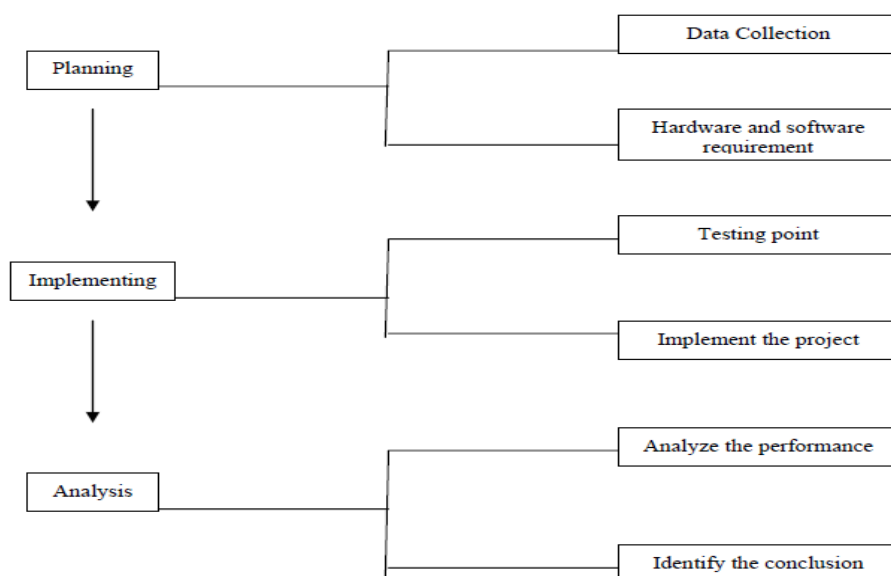


Fig 6.2 : Steps of Methodology

Planning:

To identify all the information and requirement such as hardware and software, planning must be done in the proper manner. The planning phase has two main elements namely data collection and the requirements of hardware and software.

Data collection:

Machine learning needs two things to work, data (lots of it) and models. When acquiring the data, be sure to have enough features (aspect of data that can help for a prediction, like the surface of the house to predict its price) populated to train correctly your learning model. In general, the more data you have the better so make to come with enough rows.

The primary data collected from the online sources remains in the raw form of statements, digits and qualitative terms. The raw data contains error, omissions and inconsistencies. It requires corrections after careful scrutinizing the completed questionnaires. The following steps are involved in the processing of primary data. A huge volume of raw data collected through field survey needs to be grouped for similar details of individual responses.

Data Preprocessing is a technique that is used to convert the raw data into a clean data set. In other words, whenever the data is gathered from different sources it is collected in raw format which is not feasible for the analysis.

Therefore, certain steps are executed to convert the data into a small clean data set. This technique is performed before the execution of Iterative Analysis. The set of steps is known as Data Pre-processing. It includes -

- Data Cleaning
- Data Integration
- Data Transformation
- Data Reduction

Data Preprocessing is necessary because of the presence of unformatted real-world data. Mostly real-world data is composed of -

- **Inaccurate data (missing data)** - There are many reasons for missing data such as data is not continuously collected, a mistake in data entry, technical problems with biometrics and much more.
- **The presence of noisy data (erroneous data and outliers)** - The reasons for the existence of noisy data could be a technological problem of gadget that gathers data, a human mistake during data entry and much more.
- **Inconsistent data** - The presence of inconsistencies are due to the reasons such that existence of duplication within data, human data entry, containing mistakes in codes or names, i.e., violation of data constraints and much more.

Implementing

In this work, a business intelligent model has been developed, to classify different animals, based on a specific business structure deal with Animal classification using a suitable machine learning technique. The model was evaluated by a scientific approach to measure accuracy. We are using Convolutional Neural Network (CNN) to build our model.

Analysis

In this final phase, we will test our classification model on our prepared image dataset and also measure the performance on our dataset. To evaluate the performance of our created classification and make it comparable to current approaches, we use accuracy to measure the effectiveness of classifiers.

After model building, knowing the power of model prediction on a new instance, is very important issue. Once a predictive model is developed using the historical data, one would be curious as to how the model will perform on the data that it has not seen during the model building process. One might even try multiple model types for the same prediction problem, and then, would like to know which model is the one to use for the real-world decision making situation, simply by comparing them on their prediction performance (e.g., accuracy). To measure the performance of a predictor, there are commonly used performance metrics, such as accuracy, recall etc. First, the most commonly used performance metrics will be described, and then some famous estimation methodologies are explained and compared to each other. "Performance Metrics for Predictive Modelling In classification problems, the primary source of performance

measurements is a coincidence matrix (**classification matrix or a contingency table**)". Above figure shows a coincidence matrix for a two-class classification problem. The equations of the most commonly used metrics that can be calculated from the coincidence matrix are also given in Fig 2.7.

		True Class	
		Positive	Negative
Predicted Class	Positive	True Positive Count (TP)	False Positive Count (FP)
	Negative	False Negative Count (FN)	True Negative Count (TN)

$$\text{True Positive Rate} = \frac{TP}{TP + FN}$$

$$\text{True Negative Rate} = \frac{TN}{TN + FP}$$

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

$$\text{Precision} = \frac{TP}{TP + FP}$$

$$\text{Recall} = \frac{TP}{TP + FN}$$

Figure 6.3 : confusion matrix and formulae

As being seen in above figure, the numbers along the diagonal from upper-left to lower-right represent the correct decisions made, and the numbers outside this diagonal represent the errors. "The true positive rate (also called hit rate or recall) of a classifier is estimated by dividing the correctly classified positives (the true positive count) by the total positive count. The false positive rate (also called a false alarm rate) of the classifier is estimated by dividing the incorrectly classified negatives (the false negative count) by the total negatives. The overall accuracy of a classifier is estimated by dividing the total correctly classified positives and negatives by the total number of samples.

The architecture of a ConvNet is analogous to that of the connectivity pattern of Neurons in the Human Brain and was inspired by the organization of the Visual Cortex. Individual neurons respond to stimuli only in a restricted region of the visual field known as the Receptive Field. A collection of such fields overlap to cover the entire visual area.

Flexibility

Sometimes you just don't want to use what is already there but you want to define something of your own (for example a cost function, a metric, a layer, etc.).

Although Keras 2 has been designed in such a way that you can implement almost everything you want but we all know that low-level libraries provides more flexibility. Same is the case with TF. *You can tweak* TF much more as compared to Keras.

Functionality

Although Keras provides all the general purpose functionalities for building Deep learning models, it doesn't provide as much as TF. TensorFlow offers more advanced operations as compared to Keras. This comes very handy if you are doing a research or developing some special kind of deep learning models. Some examples regarding high level operations are:

Threading and Queues

Queues are a powerful mechanism for computing tensors asynchronously in a graph. Similarly, you can execute multiple threads for the same Session for parallel computations and hence speed up your operations.

Debugger

Another extra power of TF. With TensorFlow, you get a specialized debugger. It provides visibility into the internal structure and states of running TensorFlow graphs. Insights from debugger can be used to facilitate debugging of various types of bugs during both training and inference.

Control

The more control you have over your network, more better understanding you have of what's going on with your network. With TF, you get such a control over your network. You can control whatever you want in your network. Operations on weights or gradients can be done like a charm in TF.

NumPy

NumPy, which stands for Numerical Python, is a library consisting of multidimensional array objects and a collection of routines for processing those arrays. Using NumPy, mathematical and logical operations on arrays can be performed. This tutorial explains the basics of NumPy such as its architecture and environment. It also discusses the various array functions, types of indexing, etc. An introduction to Matplotlib is also provided. All this is explained with the help of examples for better understanding.

NumPy is a Python package. It stands for 'Numerical Python'. It is a library consisting of multidimensional array objects and a collection of routines for processing of array.

Numeric, the ancestor of NumPy, was developed by Jim Hugunin. Another package Numarray was also developed, having some additional functionality. In 2005, Travis Oliphant created NumPy package by incorporating the features of Numarray into Numeric package. There are many contributors to this open source project.

Operations using NumPy

Using NumPy, a developer can perform the following operations –

- Mathematical and logical operations on arrays.
- Fourier transforms and routines for shape manipulation.
- Operations related to linear algebra. NumPy has in-built functions for linear algebra and random number generation.

NumPy A Replacement for Mat Lab

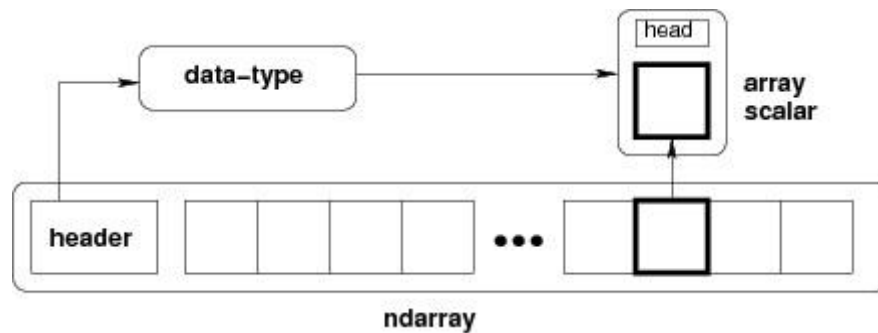
NumPy is often used along with packages like **SciPy** (Scientific Python) and **Matplotlib** (plotting library). This combination is widely used as a replacement for MatLab, a popular platform for technical computing. However, Python alternative to MatLab is now seen as a more modern and complete programming language.

It is open source, which is an added advantage of NumPy.

The most important object defined in NumPy is an N-dimensional array type called **ndarray**. It describes the collection of items of the same type. Items in the collection can be accessed using a zero-based index.

Every item in an ndarray takes the same size of block in the memory. Each element in ndarray is an object of data-type object (called **dtype**).

Any item extracted from ndarray object (by slicing) is represented by a Python object of one of array scalar types. The following diagram shows a relationship between ndarray, data type object (dtype) and array scalar type –



An instance of ndarray class can be constructed by different array creation routines described later in the tutorial. The basic ndarray is created using an array function in NumPy as follows –

```
numpy.array
```

It creates an ndarray from any object exposing array interface, or from any method that returns an array.

The ndarray objects can be saved to and loaded from the disk files. The IO functions available are –

- **load()** and **save()** functions handle /numPy binary files (with **npextension**)
- **loadtxt()** and **savetxt()** functions handle normal text files

NumPy introduces a simple file format for ndarray objects. This **.npy** file stores data, shape, dtype and other information required to reconstruct the ndarray in a disk file such that the array is correctly retrieved even if the file is on another machine with different architecture.

```
numpy.save()
```

The **numpy.save()** file stores the input array in a disk file with **npextension**.

```
import numpy as np

a = np.array([1,2,3,4,5])
```

```
np.save('outfile',a)
```

To reconstruct array from **outfile.npy**, use **load()** function.

```
import numpy as np  
  
b = np.load('outfile.npy')  
  
print b
```

It will produce the following output –

```
array([1, 2, 3, 4, 5])
```

The **save()** and **load()** functions accept an additional Boolean parameter **allow_pickle**. A pickle in Python is used to serialize and de-serialize objects before saving to or reading from a disk file.

savetxt()

The storage and retrieval of array data in simple text file format is done with **savetxt()** and **loadtxt()** functions.

Example

```
import numpy as np  
  
a = np.array([1,2,3,4,5])  
  
np.savetxt('out.txt',a)  
  
b = np.loadtxt('out.txt')  
  
print b
```

It will produce the following output –

```
[ 1.  2.  3.  4.  5.]
```

The **savetxt()** and **loadtxt()** functions accept additional optional parameters such as header, footer, and delimiter.

Pandas:

pandas is a Python package providing fast, flexible, and expressive data structures designed to make working with structured (tabular, multidimensional, potentially heterogeneous) and time series data both easy and intuitive. It aims to be the fundamental high-level building block for doing practical, **real world** data analysis in Python. Additionally, it has the broader goal of becoming **the most powerful and flexible open source data analysis / manipulation tool available in any language**. It is already well on its way toward this goal.

pandas is well suited for many different kinds of data:

- Tabular data with heterogeneously-typed columns, as in an SQL table or Excel spreadsheet
- Ordered and unordered (not necessarily fixed-frequency) time series data.
- Arbitrary matrix data (homogeneously typed or heterogeneous) with row and column labels
- Any other form of observational / statistical data sets. The data actually need not be labeled at all to be placed into a pandas data structure

The two primary data structures of pandas, Series (1-dimensional) and Data Frame (2-dimensional), handle the vast majority of typical use cases in finance, statistics, social science, and many areas of engineering. For R users, Data Frame provides everything that R's `data. Frame` provides and much more. Pandas is built on top of NumPy and is intended to integrate well within a scientific computing environment with many other 3rd party libraries.

Here are just a few of the things that pandas do well:

- Easy handling of **missing data** (represented as NaN) in floating point as well as non-floating point data
- Size mutability: columns can be **inserted and deleted** from DataFrame and higher dimensional objects
- Automatic and explicit **data alignment**: objects can be explicitly aligned to a set of labels, or the user can simply ignore the labels and let *Series*, *DataFrame*, etc. automatically align the data for you in computations
- Powerful, flexible **group by** functionality to perform split-apply-combine operations on data sets, for both aggregating and transforming data

- Make it **easy to convert** ragged, differently-indexed data in other Python and NumPy data structures into DataFrame objects
- Intelligent label-based **slicing**, **fancy indexing**, and **subsetting** of large data sets
- Intuitive **merging** and **joining** data sets
- Flexible **reshaping** and pivoting of data sets
- **Hierarchical** labeling of axes (possible to have multiple labels per tick)
- Robust IO tools for loading data from **flat files** (CSV and delimited), Excel files, databases, and saving / loading data from the ultrafast **HDF5 format**
- **Time series**-specific functionality: date range generation and frequency conversion, moving window statistics, moving window linear regressions, date shifting and lagging, etc.

Many of these principles are here to address the shortcomings frequently experienced using other languages / scientific research environments. For data scientists, working with data is typically divided into multiple stages: munging and cleaning data, analyzing / modeling it, then organizing the results of the analysis into a form suitable for plotting or tabular display. pandas is the ideal tool for all of these tasks.

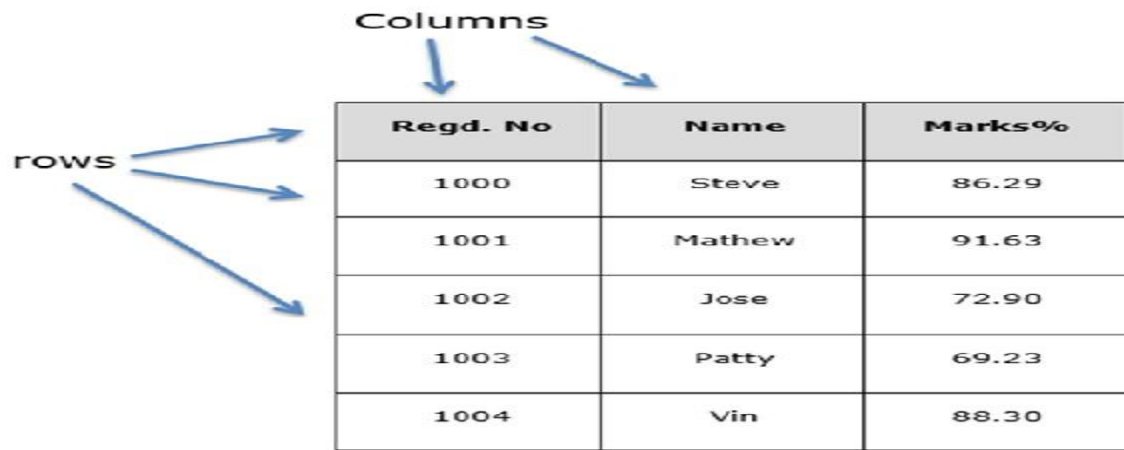
A Data frame is a two-dimensional data structure, i.e., data is aligned in a tabular fashion in rows and columns.

Features of DataFrame

- Potentially columns are of different types
- Size – Mutable
- Labeled axes (rows and columns)
- Can Perform Arithmetic operations on rows and columns

Structure

Let us assume that we are creating a data frame with student's data.



Regd. No	Name	Marks%
1000	Steve	86.29
1001	Mathew	91.63
1002	Jose	72.90
1003	Patty	69.23
1004	Vin	88.30

You can think of it as an SQL table or a spreadsheet data representation.

`pandas.DataFrame`

A pandas DataFrame can be created using the following constructor –

```
pandas.DataFrame( data, index, columns, dtype, copy)
```

The parameters of the constructor are as follows –

Sr.No	Parameter & Description
1	data data takes various forms like ndarray, series, map, lists, dict, constants and also another DataFrame.
2	index For the row labels, the Index to be used for the resulting frame is Optional Default <code>np.arange(n)</code> if no index is passed.
3	columns For column labels, the optional default syntax is - <code>np.arange(n)</code> . This is only true if no index is passed.
4	dtype Data type of each column.

5	<p>copy</p> <p>This command (or whatever it is) is used for copying of data, if the default is False.</p>
---	--

Create DataFrame

A pandas DataFrame can be created using various inputs like –

- Lists
- dict
- Series
- Numpy ndarrays
- Another DataFrame

In the subsequent sections of this chapter, we will see how to create a DataFrame using these inputs.

Create an Empty DataFrame

A basic DataFrame, which can be created is an Empty Dataframe.

Example

```
#import the pandas library and aliasing as pd

import pandas as pd

df = pd.DataFrame()

print df
```

Its **output** is as follows –

```
Empty DataFrame
Columns: []
Index: []
```

Create a DataFrame from Lists

The DataFrame can be created using a single list or a list of lists.

Example 1

```
import pandas as pd

data = [1,2,3,4,5]

df = pd.DataFrame(data)

print df
```

Its **output** is as follows –

```
0
0  1
1  2
2  3
3  4
4  5
```

Example 2

```
import pandas as pd

data = [['Alex',10],['Bob',12],['Clarke',13]]

df = pd.DataFrame(data,columns=['Name','Age'])

print df
```

Its **output** is as follows –

```
   Name  Age
0  Alex   10
1  Bob    12
2  Clarke  13
```

The **Pandas I/O API** is a set of top level reader functions accessed like **pd.read_csv()** that generally return a Pandas object.

The two workhorse functions for reading text files (or the flat files) are **read_csv()** and **read_table()**. They both use the same parsing code to intelligently convert tabular data into a **DataFrame** object –

```
pandas.read_csv(filepath_or_buffer, sep=',', delimiter=None, header='infer',
names=None, index_col=None, usecols=None
pandas.read_csv(filepath_or_buffer, sep='\t', delimiter=None, header='infer',
names=None, index_col=None, usecols=None
```

Here is how the **csv** file data looks like –

```
S.No,Name,Age,City,Salary
1,Tom,28,Toronto,20000
2,Lee,32,HongKong,3000
3,Steven,43,Bay Area,8300
4,Ram,38,Hyderabad,3900
```

Save this data as **temp.csv** and conduct operations on it.

```
S.No,Name,Age,City,Salary
1,Tom,28,Toronto,20000
2,Lee,32,HongKong,3000
3,Steven,43,Bay Area,8300
4,Ram,38,Hyderabad,3900
```

Save this data as **temp.csv** and conduct operations on it.

read.csv

read.csv reads data from the csv files and creates a DataFrame object.

```
import pandas as pd

df=pd.read_csv("temp.csv")

print df
```

Its **output** is as follows –

```
  S.No  Name  Age  City  Salary
0    1   Tom   28  Toronto  20000
```

1	2	Lee	32	HongKong	3000
2	3	Steven	43	Bay Area	8300
3	4	Ram	38	Hyderabad	3900

custom index

This specifies a column in the csv file to customize the index using **index_col**.

```
import pandas as pd

df=pd.read_csv("temp.csv",index_col=['S.No'])

print df
```

Its **output** is as follows –

S.No	Name	Age	City	Salary
1	Tom	28	Toronto	20000
2	Lee	32	HongKong	3000
3	Steven	43	Bay Area	8300
4	Ram	38	Hyderabad	3900

Converters

dtype of the columns can be passed as a dict.

```
import pandas as pd

df = pd.read_csv("temp.csv", dtype={'Salary': np.float64})

print df.dtypes
```

Its **output** is as follows –

```
S.No    int64
Name    object
Age     int64
City    object
Salary  float64
dtype: object
```

By default, the **dtype** of the Salary column is **int**, but the result shows it as **float** because we have explicitly casted the type.

Thus, the data looks like float –

```
S.No  Name  Age  City  Salary
0  1   Tom  28  Toronto  20000.0
1  2   Lee  32  HongKong  3000.0
2  3 Steven  43  Bay Area  8300.0
3  4   Ram  38  Hyderabad  3900.0
```

header_names

Specify the names of the header using the names argument.

```
import pandas as pd

df=pd.read_csv("temp.csv", names=['a', 'b', 'c','d','e'])

print df
```

Its **output** is as follows –

```
   a   b  c   d   e
0 S.No  Name  Age  City  Salary
1  1   Tom  28  Toronto  20000
2  2   Lee  32  HongKong  3000
3  3 Steven  43  Bay Area  8300
4  4   Ram  38  Hyderabad  3900
```

Observe, the header names are appended with the custom names, but the header in the file has not been eliminated. Now, we use the header argument to remove that.

If the header is in a row other than the first, pass the row number to header. This will skip the preceding rows.

```
import pandas as pd

df=pd.read_csv("temp.csv",names=['a','b','c','d','e'],header=0)

print df
```

Its **output** is as follows –

	a	b	c	d	e
0	S.No	Name	Age	City	Salary
1	1	Tom	28	Toronto	20000
2	2	Lee	32	HongKong	3000
3	3	Steven	43	Bay Area	8300
4	4	Ram	38	Hyderabad	3900

```
import pandas as pd

df=pd.read_csv("temp.csv", skiprows=2)

print df
```

Its **output** is as follows –

2	Lee	32	HongKong	3000
0	3	Steven	43	Bay Area
1	4	Ram	38	Hyderabad

Matplotlib

Matplotlib is an amazing visualization library in Python for 2D plots of arrays. Matplotlib is a multi-platform data visualization library built on NumPy arrays and designed to work with the broader SciPy stack. It was introduced by John Hunter in the year 2002. One of the greatest benefits of visualization is that it allows us visual access to huge amounts of data in easily digestible visuals. Matplotlib consists of several plots like line, bar, scatter, histogram etc.

Installation:

Windows, Linux and macOS distributions have matplotlib and most of its dependencies as wheel packages. Run the following command to install matplotlib package :

```
python -mpip install -U matplotlib
```

Importing matplotlib:

```
from matplotlib import pyplot as plt

or

import matplotlib.pyplot as plt
```

Basic plots in Matplotlib :

Matplotlib comes with a wide variety of plots. Plots helps to understand trends, patterns, and to make correlations. They're typically instruments for reasoning about quantitative information.

Uses of matplotlib

- Matplotlib is a plotting library for the Python programming language and its numerical mathematics extension NumPy. It provides an object-**oriented** API for **embedding** plots into applications using general-purpose GUI toolkits like Tkinter, wxPython, Qt, or GTK+.
- MATLAB-like interface, particularly when combined with IPython. For the power user, you have full control of line styles, font properties, axes properties, etc, via an object oriented interface or via a set of functions familiar to MATLAB users.
- Matplotlib is a Python package for 2D plotting that generates production-quality graphs. It supports interactive and non-interactive plotting, and can save images in several output formats (PNG, PS, and others). It can use multiple window toolkits (GTK+, wxWidgets, Qt, and so on) and it provides a wide variety of plot types (lines, bars, pie charts, histograms, and many more).
- In addition to this, it is highly customizable, flexible, and easy to use. The dual nature of Matplotlib allows it to be used in both interactive and non-interactive scripts. It can be used in scripts without a graphical display, embedded in graphical applications, or on web pages. It can also be used interactively with the Python interpreter or IPython.

Merits of Matplotlib

- The idea behind Matplotlib can be summed up in the following motto as quoted by John Hunter, the creator and project leader of Matplotlib:
- “Matplotlib tries to make easy things easy and hard things possible”.
- Matplotlib was born in the scientific area of computing, where gnuplot and MATLAB were (and still are) used a lot.

- Matplotlib was modeled on MATLAB, because graphing was something that MATLAB did very well. The high degree of compatibility between them made many people move from MATLAB to Matplotlib, as they felt like home while working with Matplotlib.
- But what are the points that built the success of Matplotlib? Let's look at some of them:
- **It uses Python:** Python is a very interesting language for scientific purposes (it's interpreted, high-level, easy to learn, easily extensible, and has a powerful standard library) and is now used by major institutions such as NASA, JPL, Google, DreamWorks, Disney, and many more.
- **It's open source, so no license to pay:** This makes it very appealing for professors and students, who often have a low budget.
- **It's a real programming language:** The MATLAB language (while being Turing-complete) lacks many of the features of a general-purpose language like Python.
- **It's much more complete:** Python has a lot of external modules that will help us perform all the functions we need to. So it's the perfect tool to acquire data, elaborate the data, and then plot the data.
- **It's very customizable and extensible:** Matplotlib can fit every use case because it has a lot of graph types, features, and configuration options.
- **It's integrated with LaTeX markup:** This is really useful when writing scientific papers.
- **It's cross-platform and portable:** Matplotlib can run on Linux, Windows, Mac OS X, and Sun Solaris (and Python can run on almost every architecture available).
- The aim of Matplotlib is to generate graphs. So, we need a way to actually view these images or even to save them to files. We're going to look at the various output formats available in Matplotlib and the graphical user interfaces (GUIs) supported by the library.
- Matplotlib supports both the categories, particularly with the following output formats:

Format	Type Description	Description
EPS	Vector	Encapsulated PostScript
JPG	Raster	Graphic format with lossy compression method for

		photographic output.
PDF	Vector	Portable Document Format (PDF).
PNG	Raster	Portable Network Graphics (PNG), a raster graphics format with a lossless compression method (more adaptable to line art than JPG).
PS	Vector	Language widely used in publishing and as printers jobs format.
SVG	Vector	Scalable Vector Graphics (SVG), XML based.

Matplotlib is a plotting library for Python. It is used along with NumPy to provide an environment that is an effective open source alternative for MatLab. It can also be used with graphics toolkits like PyQt and wxPython.

Matplotlib module was first written by John D. Hunter. Since 2012, Michael Droettboom is the principal developer. Currently, Matplotlib ver. 1.5.1 is the stable version available. The package is available in binary distribution as well as in the source code form on www.matplotlib.org.

Conventionally, the package is imported into the Python script by adding the following statement –

```
from matplotlib import pyplot as plt
```

Here **pyplot()** is the most important function in matplotlib library, which is used to plot 2D data. The following script plots the equation $y = 2x + 5$

Example

```
import numpy as np

from matplotlib import pyplot as plt

x = np.arange(1,11)

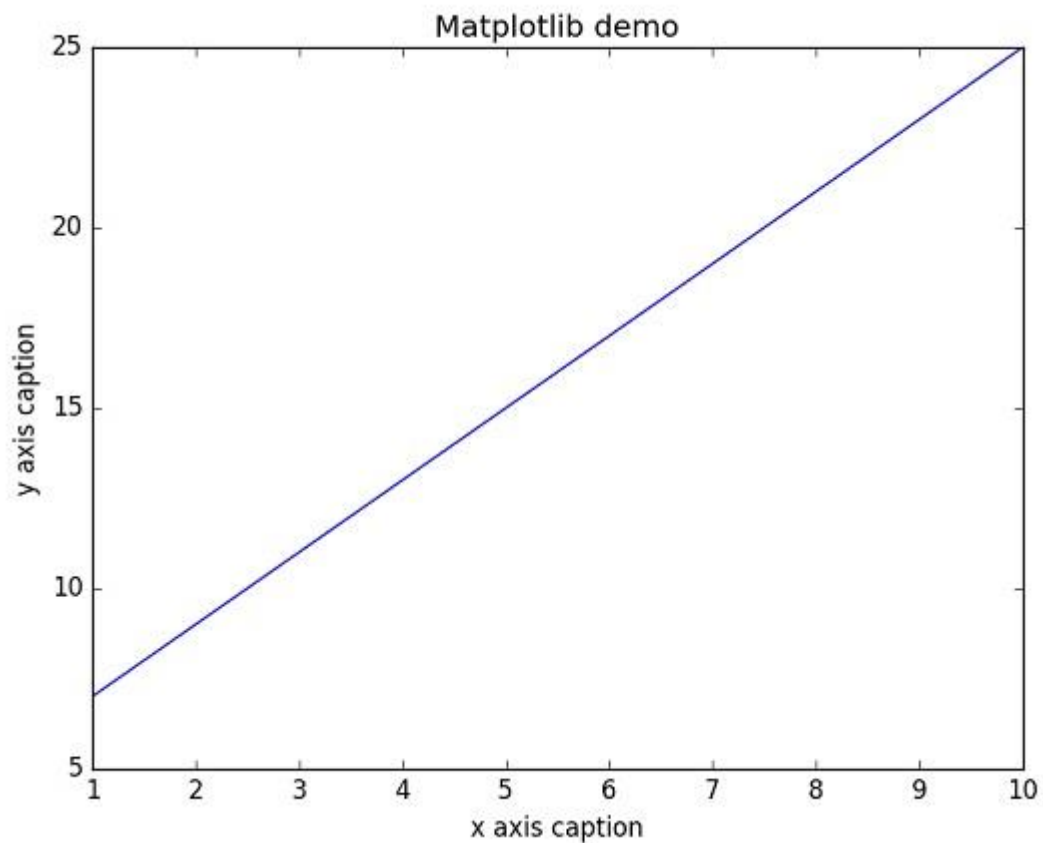
y = 2 * x + 5
```

```
plt.title("Matplotlib demo")  
  
plt.xlabel("x axis caption")  
  
plt.ylabel("y axis caption")  
  
plt.plot(x,y)  
  
plt.show()
```

An ndarray object **x** is created from **np.arange()** function as the values on the **x axis**. The corresponding values on the **y axis** are stored in another **ndarray object y**. These values are plotted using **plot()** function of pyplot submodule of matplotlib package.

The graphical representation is displayed by **show()** function.

The above code should produce the following output –



Instead of the linear graph, the values can be displayed discretely by adding a format string to the **plot()** function. Following formatting characters can be used.

NumPy has a **numpy.histogram()** function that is a graphical representation of the frequency distribution of data. Rectangles of equal horizontal size corresponding to class interval called **bin** and **variable height** corresponding to frequency.

`numpy.histogram()`

The `numpy.histogram()` function takes the input array and bins as two parameters. The successive elements in bin array act as the boundary of each bin.

```
import numpy as np

a = np.array([22,87,5,43,56,73,55,54,11,20,51,5,79,31,27])

np.histogram(a,bins = [0,20,40,60,80,100])

hist,bins = np.histogram(a,bins = [0,20,40,60,80,100])

print hist

print bins
```

It will produce the following output –

```
[3 4 5 2 1]
[0 20 40 60 80 100]
```

`plt()`

Matplotlib can convert this numeric representation of histogram into a graph. The **plt()** **function** of pyplot submodule takes the array containing the data and bin array as parameters and converts into a histogram.

```
from matplotlib import pyplot as plt

import numpy as np

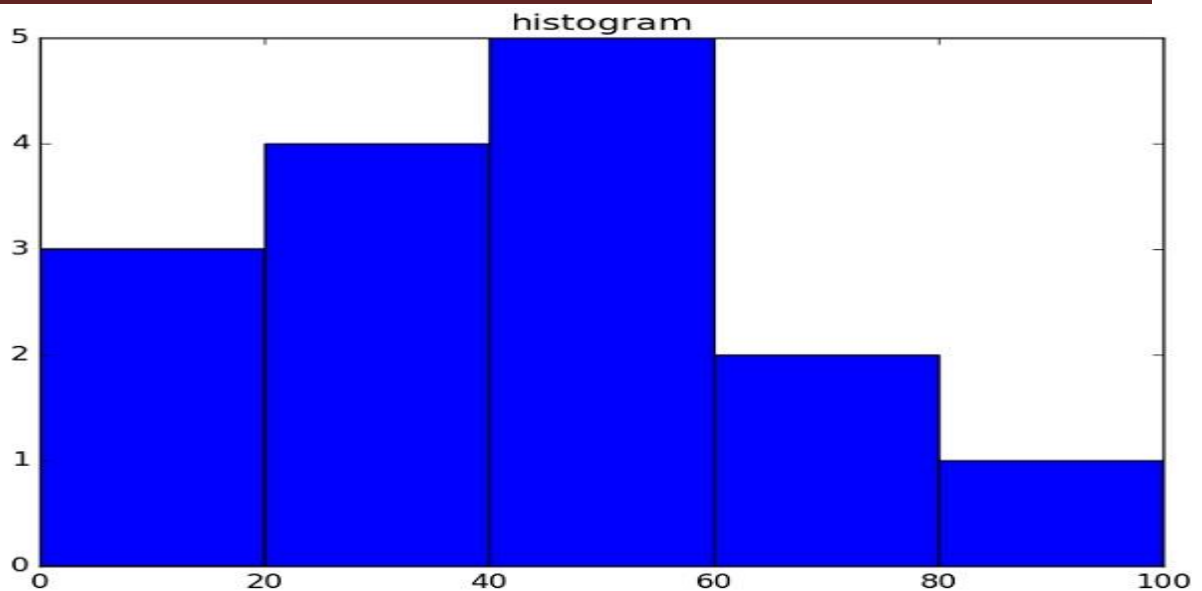
a = np.array([22,87,5,43,56,73,55,54,11,20,51,5,79,31,27])

plt.hist(a, bins = [0,20,40,60,80,100])

plt.title("histogram")

plt.show()
```

It should produce the following output –



Backends

- A backend that displays the image on screen is called a user interface backend.
- The backend is that part of Matplotlib that works behind the scenes and allows the software to target several different output formats and GUI libraries (for screen visualization).
- In order to be even more flexible, Matplotlib introduces the following two layers structured (only for GUI output):
- **The renderer:** This actually does the drawing
- **The canvas:** This is the destination of the figure.
- The standard renderer is the Anti-Grain Geometry (AGG) library, a high performance rendering engine which is able to create images of publication level quality, with anti-aliasing, and sub pixel accuracy. AGG is responsible for the beautiful appearance of Matplotlib graphs.
- The canvas is provided with the GUI libraries, and any of them can use the AGG rendering, along with the support for other rendering engines (for example, GTK+).
- Let's have a look at the user interface toolkits and their available renderers:

Backend	Description
GTKAgg	GTK+ (The GIMP ToolKit GUI library) canvas with AGG rendering.

GTK	GTK+ canvas with GDK rendering. GDK rendering is rather primitive, and doesn't include anti-aliasing for the smoothing of lines.
GTKCairo	GTK+ canvas with Cairo rendering.
WxAgg	wxWidgets (cross-platform GUI and tools library for GTK+, Windows, and Mac OS X. It uses native widgets for each operating system, so applications will have the look and feel that users expect on that operating system) canvas with AGG rendering.
WX	wxWidgets canvas with native wxWidgets rendering.
TkAgg	TkAgg Tk (graphical user interface for Tcl and many other dynamic languages) canvas with AGG rendering.

S K Learn

Introduction

scikit-learn is a library, i.e. a collection of classes and functions that users import into Python programs. Using scikit-learn therefore requires basic Python programming knowledge. No command-line interface, let alone a graphical user interface, is offered for non-programmer users

Scikit-learn is the most useful library for machine learning in Python. It is on NumPy, SciPy and matplotlib, this library contains a lot of efficient tools for machine learning and statistical modeling including classification, regression, clustering and dimensionality reduction.

Scikit-learn provide a range of supervised and unsupervised learning algorithms via a consistent interface in Python. It is licensed under a permissive simplified BSD license and is distributed under many Linux distributions, encouraging academic and commercial use.

Scikit-learn is an open source Python library that has powerful tools for data analysis and data mining. It's available under the BSD license and is built on the following machine learning libraries: NumPy, a library for manipulating multi-dimensional arrays and matrices.

Core API

All objects within scikit-learn share a uniform common basic API consisting of three complementary interfaces: an estimator interface for building and fitting models, a

predictor interface for making predictions and a transformer interface for converting data. In this section, we describe these three interfaces, after reviewing our general principles and data representation choices.

General principles As much as possible, our design choices have been guided so as to avoid the proliferation of framework code. We try to adopt simple conventions and to limit to a minimum the number of methods an object must implement. The API is designed to adhere to the following broad principles:

Consistency: All objects (basic or composite) share a consistent interface composed of a limited set of methods. This interface is documented in a consistent manner for all objects.

Inspection: Constructor parameters and parameter values determined by learning algorithms are stored and exposed as public attributes.

Non-proliferation of classes: Learning algorithms are the only objects to be represented using custom classes. Datasets are represented as NumPy arrays or SciPy sparse matrices. Hyper-parameter names and values are represented as standard Python strings or numbers whenever possible. This keeps scikit learn easy to use and easy to combine with other libraries.

Composition: Many machine learning tasks are expressible as sequences or combinations of transformations to data. Some learning algorithms are also naturally viewed as meta-algorithms parameterized on other algorithms. Whenever feasible, such algorithms are implemented and composed from existing building blocks. Sensible defaults. Whenever an operation requires a user-defined parameter, an appropriate default value is defined by the library. The default value should cause the operation to be performed in a sensible way (giving a baseline solution for the task at hand).

Components of scikit-learn:

Scikit-learn comes loaded with a lot of features. Here are a few of them to help you understand the spread:

- **Supervised learning algorithms:** Think of any supervised learning algorithm you might have heard about and there is a very high chance that it is part of scikit-learn. Starting from Generalized linear models (e.g Linear Regression), Support Vector Machines (SVM), Decision Trees to Bayesian methods – all of them are

part of scikit-learn toolbox. The spread of algorithms is one of the big reasons for high usage of scikit-learn. I started using scikit to solve supervised learning problems and would recommend that to people new to scikit / machine learning as well.

- **Cross-validation:** There are various methods to check the accuracy of supervised models on unseen data.
- **Unsupervised learning algorithms:** Again there is a large spread of algorithms in the offering – starting from clustering, factor analysis, principal component analysis to unsupervised neural networks.
- **Various toy datasets:** This came in handy while learning scikit-learn. I had learnt SAS using various academic datasets (e.g. IRIS dataset, Boston House prices dataset). Having them handy while learning a new library helped a lot.
- **Feature extraction:** Useful for extracting features from images and text (e.g. Bag of words).

Estimators

The estimator interface is at the core of the library. It defines instantiation mechanisms of objects and exposes a fit method for learning a model from training data. All supervised and unsupervised learning algorithms (e.g., for classification, regression or clustering) are offered as objects implementing this interface. Machine learning tasks like feature extraction, feature selection or dimensionality reduction are also provided as estimators.

Predictor

The predictor interface extends the notion of an estimator by adding a predict method that takes an array X test and produces predictions for X test, based on the learned parameters of the estimator (we call the input to predict “X test” in order to emphasize that predict generalizes to new data). In the case of supervised learning estimators, this method typically returns the predicted labels or values computed by the model.

Transformers

Since it is common to modify or filter data before feeding it to a learning algorithm, some estimators in the library implement a transformer interface which defines a transform method. It takes as input some new data X test and yields as output a **transformed**

version of X test. Preprocessing, feature selection, feature extraction and dimensionality reduction algorithms are all provided as transformers within the library.

Advanced API

The advanced API mechanisms for building meta-estimators, composing complex estimators and selecting models

Meta-estimators

Some machine learning algorithms are expressed naturally as meta-algorithms parameterized on simpler algorithms. . Examples include ensemble methods which build and combine several simpler models (e.g., decision trees), or multiclass and multi label classification schemes which can be used to turn a binary classifier into a multiclass or multi label classifier.

Pipelines and feature unions

A distinguishing feature of the scikit-learn API is its ability to compose new estimators from several base estimators. Composition mechanisms can be used to combine typical machine learning workflows into a single object which is itself an estimator, and can be employed wherever usual estimators can be used.

Extending scikit-learn

To ease code reuse, simplify implementation and skip the introduction of superfluous classes, the Python principle of duck typing is exploited throughout the code base.

SYSTEM TESTING

Testing is an important phase in the development life cycle of the product. This is the phase where all the error remaining in all the phases will be detected. Hence testing plays a very critical role for quality assurance and ensuring the reliability of the software. During the test, the program to be tested is executed with a set of test cases and the output of the program for n tests cases is evaluated to determine whether the program is performing as expected.

Testing of software or hardware is conducted on complete system to evaluate its compliance with specified requirement. System testing is performed on entire system in the context of functional requirements specification and/or system requirement specification. Testing is an investigatory phase, where focus is to have almost a destructive attitude and test not only the design, but also the behavior and even the believed expectation of the customer.

7.1 TESTING OBJECTIVES

The testing objectives are as follows:

- Testing is the process of executing the program with the intent of finding an error.
- A good test case is one that has a high probability of finding an error.
- Testing cannot show the absence of defects.

7.2 TEST TYPES

Different types of tests are mentioned below

- Unit testing
- Integration testing
- System testing
 - Validation testing
 - Black Box Testing
 - White Box testing
- Acceptance testing

7.3 UNIT TESTING

Unit testing is a level of software testing where individual units/ components of a software are tested. The purpose is to validate that each unit of the software performs as designed. A unit is the smallest testable part of any software. It usually has one or a few inputs and usually a single output. In procedural programming, a unit may be an individual program, function, procedure, etc. In object-oriented programming, the smallest unit is a method, which may belong to a base/ super class, abstract class or derived/ child class. (Some treat a module of an application as a unit. This is to be discouraged as there will probably be many individual units within that module.) Unit testing frameworks, drivers, stubs, and mock/ fake objects are used to assist in unit testing.

7.4 INTEGRATION TESTING

Integration testing is a level of software testing where individual units are combined and tested as a group. The purpose of this level of testing is to expose faults in the interaction between integrated units. Test drivers and test stubs are used to assist in Integration Testing. Testing performed to expose defects in the interfaces and in the interactions between integrated components or systems.

7.5 SYSTEM TESTING

System Testing is the testing of a complete and fully integrated software product. Usually, software is only one element of a larger computer-based system. Ultimately, software is interfaced with other software/hardware systems. System Testing is actually a series of different tests whose sole purpose is to exercise the full computer-based system.

7.5.1 VALIDATION TESTING

Validation Testing, carried out by QA professionals, is to determine if the system complies with the requirements and performs functions for which it is intended and meets the organization's goals and user needs.

This kind of testing is very important, as well as verification testing. Validation is done at the end of the development process and takes place after verification is completed. Thus, to ensure customer satisfaction, developers apply validation testing.

Its goal is to validate and be confident about the product or system and that it fulfils the requirements given by the customer. The acceptance of the software from the end customer is also its part. There is a notion as Independent Validation testing – If the validation tests are carried out by a third party, they are known as independent validation and verification (IV&V). The developer needs to provide the user manual to the third party tester. This manual should clearly contain the standard working conditions of the software. These third-party organizations submit a validation report to the developer after the software is tested. The developer, upon receipt of this report, makes the required changes to the software and repeats tests it to check whether the customer needs are met or not.

Software validation testing is an important part of the software development lifecycle (SDLC), apart from verification, debugging, and certification. Validation testing ensures that the software meets the quality standards set by the customer and that the product meets customer requirements.

7.5.2 BLACK BOX TESTING

Black box testing is defined as a testing technique in which functionality of the Application Under Test (AUT) is tested without looking at the internal code structure, implementation details and knowledge of internal paths of the software. This type of testing is based entirely on software requirements and specifications. In Black Box Testing we just focus on inputs and output of the software system without bothering about internal knowledge of the software program.

Here are the generic steps followed to carry out any type of Black Box Testing.

- Initially, the requirements and specifications of the system are examined.
- Tester determines expected outputs for all those inputs.
- Software tester constructs test cases with the selected inputs.
- The test cases are executed.
- Software tester compares the actual outputs with the expected outputs.
- Defects if any are fixed and re-tested.

7.5.3 WHITE BOX TESTING

White Box Testing is defined as the testing of a software solution's internal structure, design, and coding. In this type of testing, the code is visible to the tester. It focuses

primarily on verifying the flow of inputs and outputs through the application, improving design and usability, strengthening security. White box testing is also known as Clear Box testing, Open Box testing, Structural testing, Transparent Box testing, Code-Based testing, and Glass Box testing. It is usually performed by developers.

White box testing is based on the inner workings of an application and revolves around internal testing. The term "White Box" was used because of the see-through box concept. The clear box or White Box name symbolizes the ability to see through the software's outer shell (or "box") into its inner workings.

White box testing involves the testing of the software code for the following:

- Internal security holes
- Broken or poorly structured paths in the coding processes
- The flow of specific inputs through the code
- Expected output
- The functionality of conditional loops
- Testing of each statement, object, and function on an individual basis

The testing can be done at system, integration and unit levels of software development. One of the basic goals of white box testing is to verify a working flow for an application. It involves testing a series of predefined inputs against expected or desired outputs so that when a specific input does not result in the expected output, you have encountered a bug.

7.6 ACCEPTANCE TESTING

Acceptance testing is a level of software testing where a system is tested for acceptability. Acceptance Testing is the fourth and last level of software testing performed after System Testing and before making the system available for actual use. The purpose of this test is to evaluate the system's compliance with the business requirements and assess whether it is acceptable for delivery. Firstly, the basic tests are executed, and if the test results are satisfactory then the execution of more complex scenarios are carried out. Formal testing with respect to user needs, requirements, and business processes conducted to determine whether or not a system satisfies the acceptance criteria and to enable the user, customers or other authorized entity to determine whether or not to accept the system.

Chapter 8

RESULTS

A result is the final consequence of actions or events expressed qualitatively or quantitatively. Performance analysis is an operational analysis, is a set of basic quantitative relationship between the performance quantities.

SNAPSHOTS :

The screenshot shows a Jupyter Notebook interface with the title 'main_model2'. The code cell 'load dataset' contains two lines of Python code: `st_data = pd.read_excel('data.xlsx')` and `st_data.head()`. The output of the second line is a table with 18 columns and 5 rows of data.

	Subject1	Subject2	Subject3	Subject4	Subject5	Subject6	Subject7	Subject8	Subject9	Aptitude Test	Technical Skills	English Skills	olympiads	reading and writing skills	memory capability score	Su
0	69	63	78	87	94	94	87	84	61	4	4	8	yes	excellent	excellent	I
1	78	62	73	60	71	70	73	84	91	7	2	3	no	poor	medium	Adm
2	71	86	91	87	61	81	72	72	94	1	1	3	yes	poor	excellent	Adm
3	76	87	60	84	89	73	62	88	69	1	2	5	no	medium	excellent	Adm
4	92	62	90	67	71	89	73	71	73	5	6	3	no	poor	excellent	

Snapshot 1 : The above figure says we are loading the dataset manually in .xlsx format and it shows results in table format

The screenshot shows a Jupyter Notebook interface with the title 'main_model2'. The code cell 'data preprocessing' contains several lines of Python code for data manipulation. The output of the first line shows the unique values of the 'olympiads' column, and the output of the second line shows the unique values of the 'reading and writing skills' column.

```
In [7]: st_data['olympiads'].unique()
Out[7]: array(['yes', 'no'], dtype=object)

In [8]: st_data['olympiads'].replace('no',0,inplace=True)
st_data['olympiads'].replace('yes',1,inplace=True)

In [9]: st_data['reading and writing skills'].unique()
Out[9]: array(['excellent', 'poor', 'medium'], dtype=object)

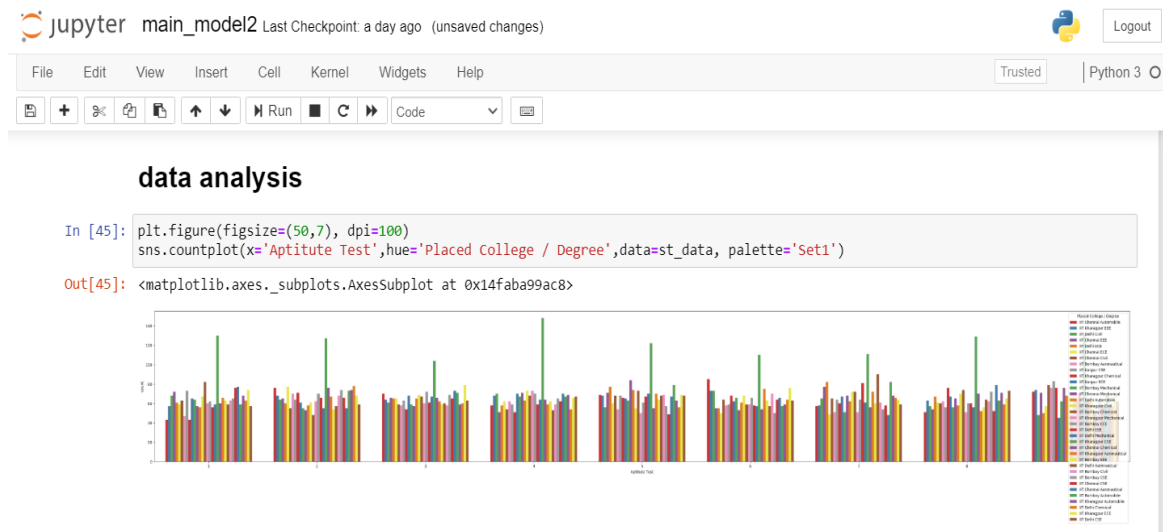
In [10]: st_data['reading and writing skills'].replace('poor',0,inplace=True)
st_data['reading and writing skills'].replace('medium',1,inplace=True)
st_data['reading and writing skills'].replace('excellent',2,inplace=True)

In [11]: st_data['memory capability score'].unique()
Out[11]: array(['excellent', 'medium', 'poor'], dtype=object)

In [12]: st_data['memory capability score'].replace('poor',0,inplace=True)
st_data['memory capability score'].replace('medium',1,inplace=True)
st_data['memory capability score'].replace('excellent',2,inplace=True)

In [13]: st_data['Suggested Job Role'].unique()
Out[13]: array(['Database Developer', 'Portal Administrator', 'Systems Security Administrator', 'Business Systems Analyst', 'Software Systems Engineer', 'Business Intelligence Analyst', 'CRM Technical Developer', 'Mobile Applications Developer', 'UX Designer', 'Quality Assurance Associate', 'Web Developer'])
```

Snapshot 2 : Here we are preprocessing the data before analyzing the data.



Snapshot 3 : In this picture we analysing the aptitude test results data using matplotlib function for bar graph representation

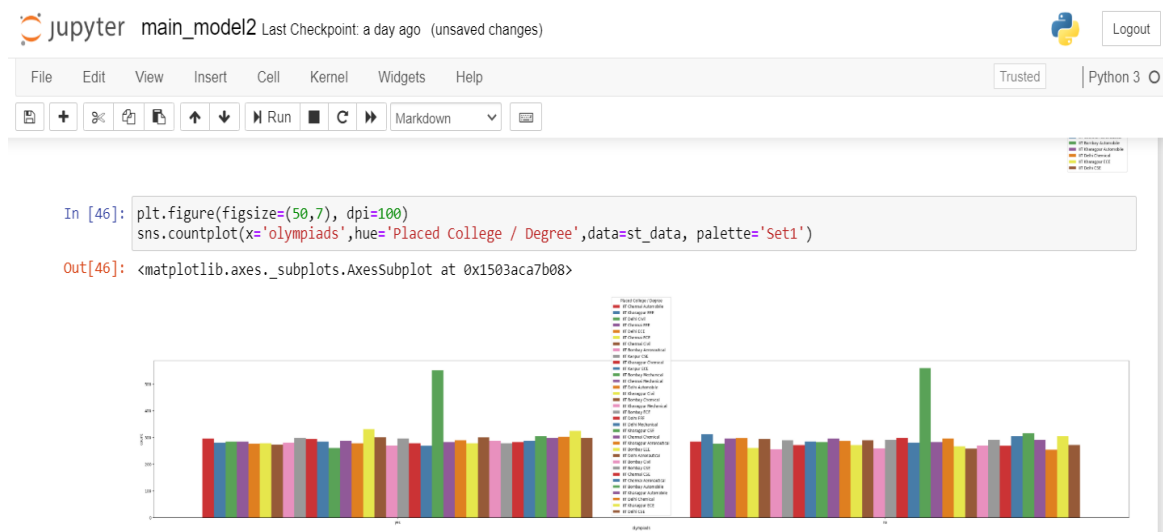


Figure 8.4 : In this picture we analyzing the olympiads data using matplotlib function for bar graph representation

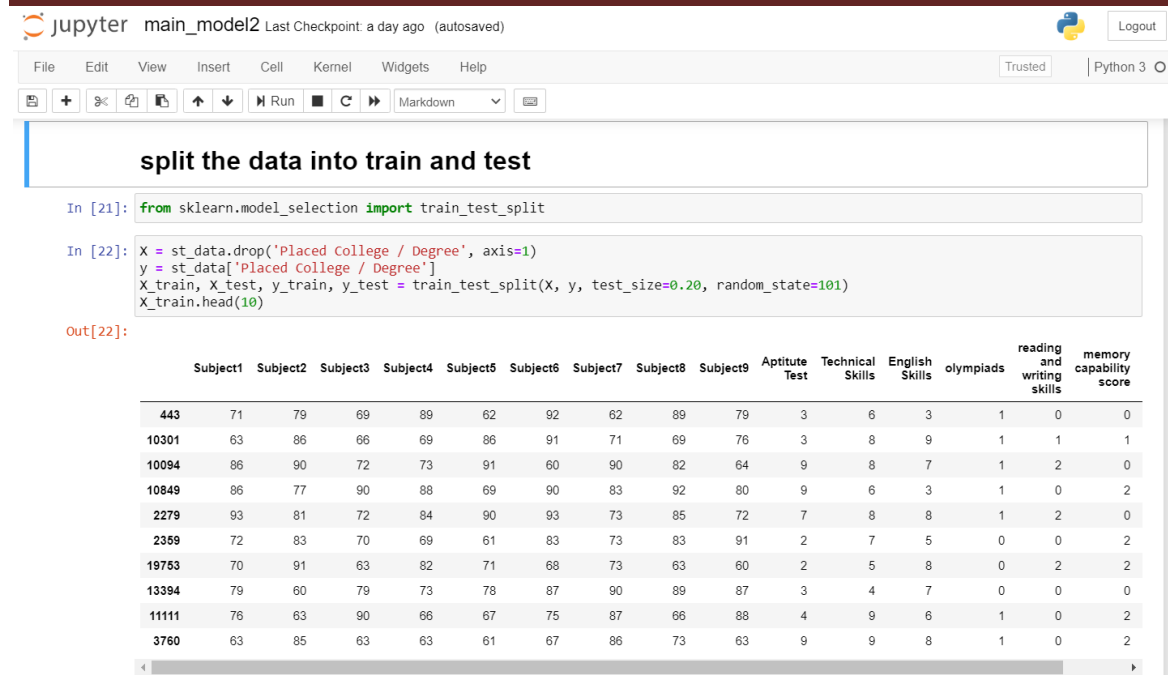


Figure 8.5 : Here we split the dataset into train and test the data

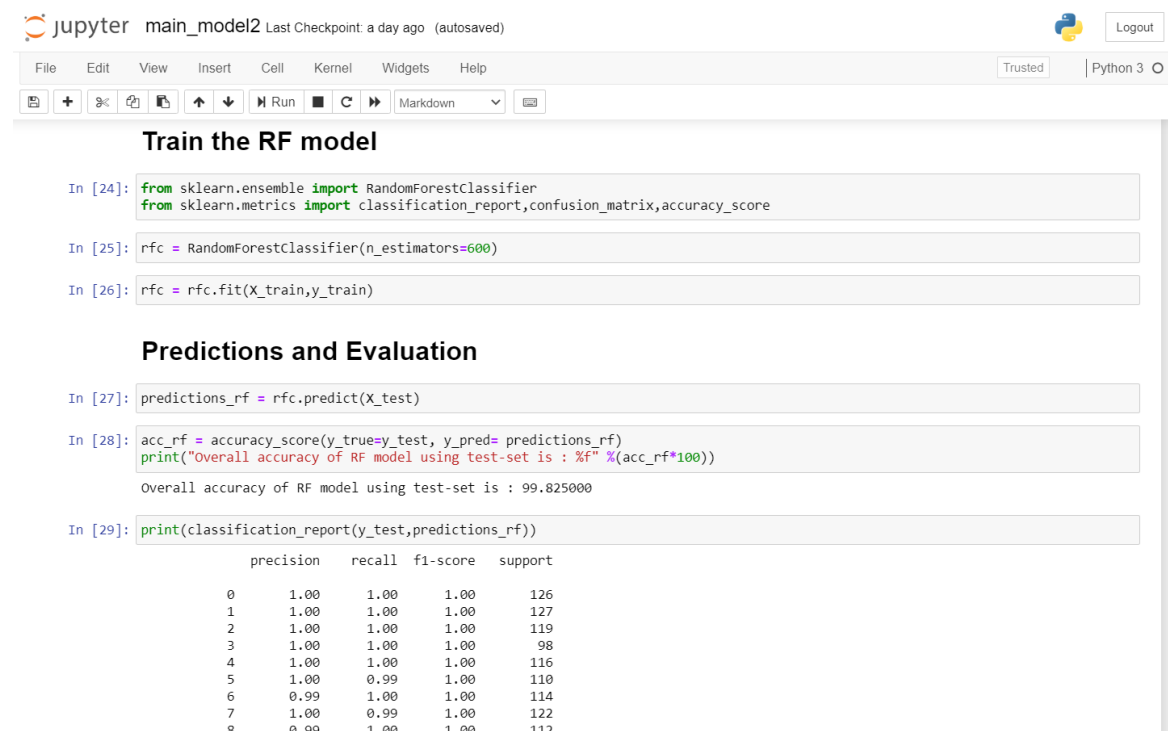
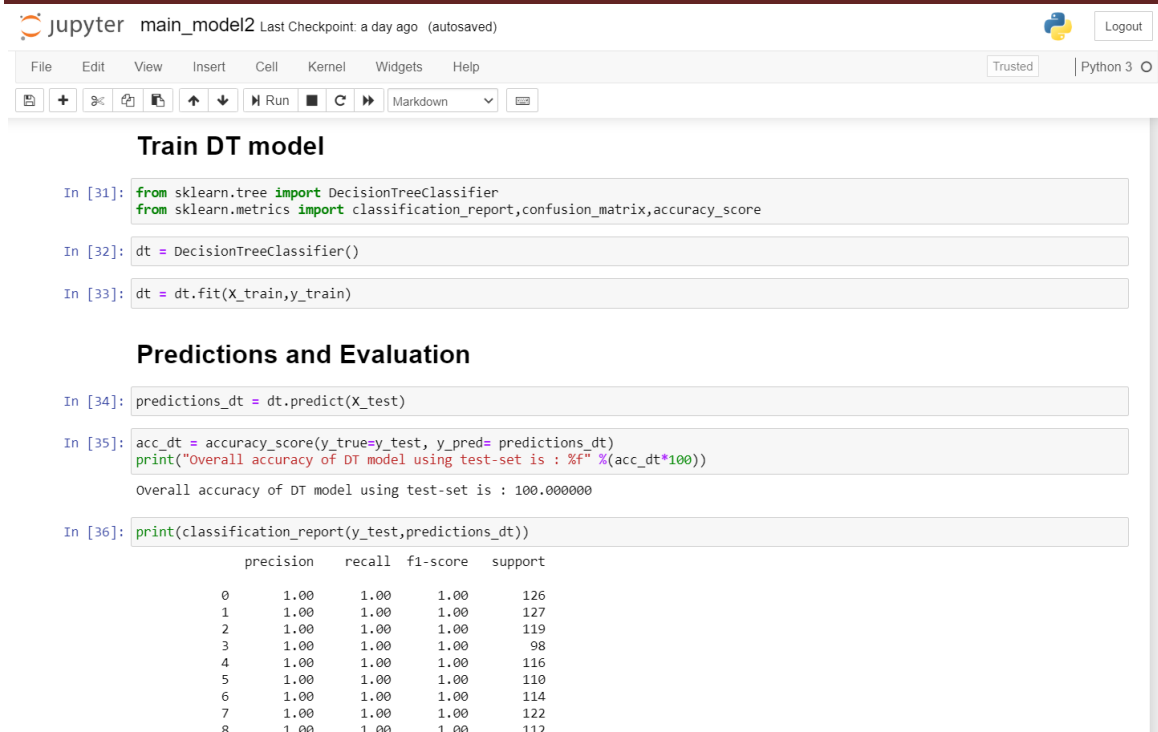


Figure 8.6 : Training the model using random forest algorithm, evaluate the data and predict the value.



The Jupyter Notebook interface shows the following code and output:

```
In [31]: from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import classification_report, confusion_matrix, accuracy_score

In [32]: dt = DecisionTreeClassifier()

In [33]: dt = dt.fit(X_train, y_train)
```

Train DT model

Predictions and Evaluation

```
In [34]: predictions_dt = dt.predict(X_test)

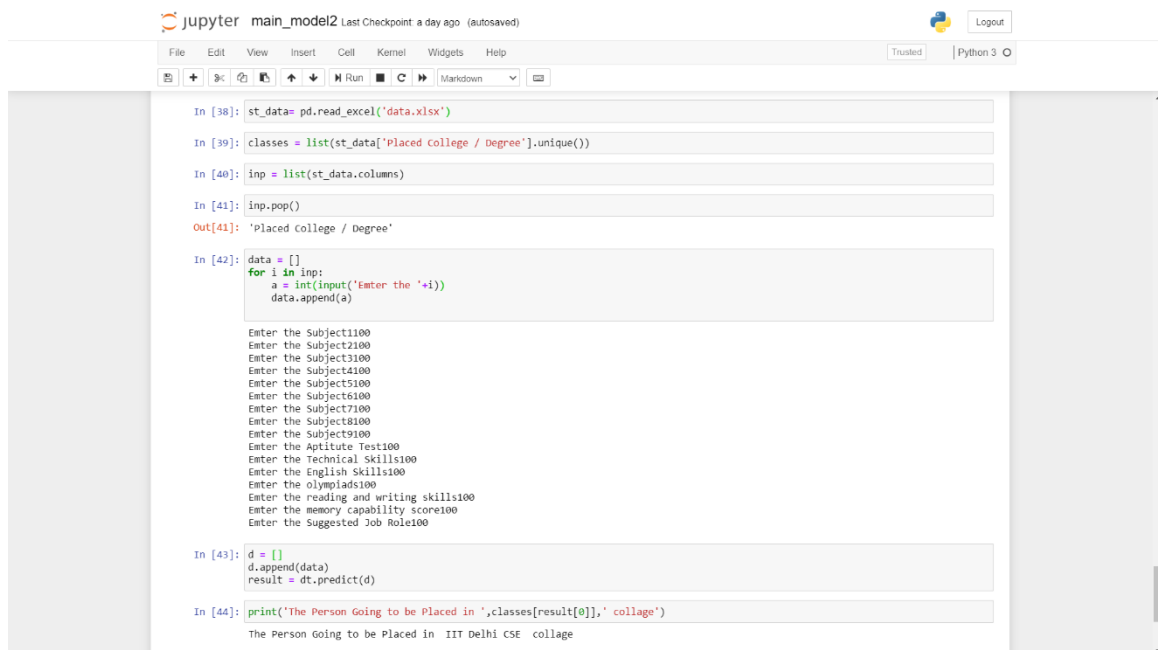
In [35]: acc_dt = accuracy_score(y_true=y_test, y_pred=predictions_dt)
print("Overall accuracy of DT model using test-set is : %f" %(acc_dt*100))

Overall accuracy of DT model using test-set is : 100.000000

In [36]: print(classification_report(y_test, predictions_dt))
```

	precision	recall	f1-score	support
0	1.00	1.00	1.00	126
1	1.00	1.00	1.00	127
2	1.00	1.00	1.00	119
3	1.00	1.00	1.00	98
4	1.00	1.00	1.00	116
5	1.00	1.00	1.00	110
6	1.00	1.00	1.00	114
7	1.00	1.00	1.00	122
8	1.00	1.00	1.00	112

Figure 8.7 : Training the model using DT algorithm, evaluate the data and predict the accuracy value.



The Jupyter Notebook interface shows the following code and output:

```
In [38]: st_data= pd.read_excel('data.xlsx')

In [39]: classes = list(st_data['Placed college / Degree'].unique())

In [40]: inp = list(st_data.columns)

In [41]: inp.pop()
Out[41]: 'Placed college / Degree'
```

```
In [42]: data = []
for i in inp:
    a = int(input('Enter the '+i))
    data.append(a)

Enter the Subject1100
Enter the Subject2100
Enter the Subject3100
Enter the Subject4100
Enter the Subject5100
Enter the Subject6100
Enter the Subject7100
Enter the Subject8100
Enter the Subject9100
Enter the Aptitude Test100
Enter the Technical Skills100
Enter the English Skills100
Enter the olympiads100
Enter the reading and writing skills100
Enter the memory capability score100
Enter the Suggested Job Role100
```

```
In [43]: d = []
d.append(data)
result = dt.predict(d)

In [44]: print('The Person Going to be Placed in ', classes[result[0]], ' collage')

The Person Going to be Placed in IIT Delhi CSE collage
```

Figure 8.8 : Here we can predict the accuracy for manual data

CONCLUSION

We find that the best model for predicting the success, whether a student will pass their college course with a better marks of a student differs between our datasets best prediction model is the Random Forest with an accuracy of 99.82% and better precision, recall, and support values than the other models. Using Decision tree algorithm at an accuracy of 100% and likewise better precision, recall, and support values than the other models

Interestingly, all of the models we utilized outperformed the current standards by approximately 200%. We have proven that machine learning models are more capable and accurate than current placement standards. Using these models will increase the accuracy and precision of student placement in either remedial or transfer-level coursework. Employing the models will decrease time and funding wasted by both the student and college. Finally, these models provide a chance for more students to be able to stand on their own abilities throughout the admission process.

REFERENCE

- Decision Support System for Admission in Engineering Colleges based on Entrance Exam Marks
- Data | Free Full-Text | Predictive Models of Student College Commitment Decisions Using Machine Learning | HTML
- Machine Learning to Predict College Course Success
- <https://research.ijcaonline.org>
- <http://www.ijarse.com>
- <https://scholar.smu.edu>
- <https://www.mdpi.com>
- <https://google.com>
- <https://youtube.com>
- <https://python.org>