

A MAJOR PROJECT REPORT
ON
“EMOTION-RESPONSIVE MUSIC SUGGESTION SYSTEM”

Submitted to

SRI INDU COLLEGE OF ENGINEERING AND TECHNOLOGY, HYDERABAD

In partial fulfillment of the requirements for the award of degree of

BACHELOR OF TECHNOLOGY

In

COMPUTER SCIENCE AND INFORMATION TECHNOLOGY

Submitted by

| | |
|-----------------------|---------------------|
| B. SAI SATWIK | [20D41A3313] |
| G. SAI RUDRESH | [20D41A3321] |
| K. SANJAY | [20D41A3332] |
| B. NIKITHA | [20D41A3308] |

Under the esteemed guidance of

Mrs. K. VIJAYALAKSHMI

(Assistant Professor)



DEPARTMENT OF COMPUTER SCIENCE AND INFORMATION TECHNOLOGY

SRI INDU COLLEGE OF ENGINEERING AND TECHNOLOGY

(An Autonomous Institution under UGC, Accredited by NBA, Affiliated to JNTUH)

Sheriguda (V), Ibrahimpatnam (M), Rangareddy Dist – 501 510

(2023-2024)

SRI INDU COLLEGE OF ENGINEERING AND TECHNOLOGY

(An Autonomous Institution under UGC, Accredited by NBA, Affiliated to JNTUH)

DEPARTMENT OF COMPUTER SCIENCE AND INFORMATION TECHNOLOGY



CERTIFICATE

Certified that the Major project entitled **“EMOTION-RESPONSIVE MUSIC SUGGESTION SYSTEM”** is a bonafide work carried out by **B. SAI SATWIK [20D41A3313]**, **G. SAI RUDRESH [20D41A3321]**, **K. SANJAY [20D41A3332]**, **B. NIKITHA [20D41A3308]** in partial fulfillment for the award of degree of Bachelor of Technology in **Computer Science and Information Technology** of SICET, Hyderabad for the academic year **2023-2024**. The project has been approved as it satisfies academic requirements in respect of the work prescribed for **IV Year, II-Semester of B. Tech** course.

INTERNAL GUIDE
(Mrs. K. VIJAYALAKSHMI)
(Assistant Professor)

HEAD OF THE DEPARTMENT
(Dr.T. CHARAN SINGH)

EXTERNAL EXAMINER

ACKNOWLEDGEMENT

The satisfaction that accompanies the successful completion of the task would be put incomplete without the mention of the people who made it possible, whose constant guidance and encouragement crown all the efforts with success. We are thankful to Principal **Dr. G. SURESH** for giving us the permission to carry out this project. We are highly indebted to **Dr. T. CHARAN SINGH, Head of the Department** of Computer Science and Information Technology, for providing necessary infrastructure and labs and also valuable guidance at every stage of this project. We are grateful to our internal project guide **Mr. K. VIJAYALAKSHMI , Assistant Professor** for her constant motivation and guidance given by her during the execution of this project work. We would like to thank the Teaching & Non-Teaching staff of Department of Computer Science and Information Technology for sharing their knowledge with us, last but not least we express our sincere thanks to everyone who helped directly or indirectly for the completion of this project.

| | |
|----------------------|---------------------|
| B. SAI SATWIK | [20D41A3313] |
| G.SAI RUDRESH | [20D41A3321] |
| K. SANJAY | [20D41A3332] |
| B.NIKITHA | [20D41A3308] |

ABSTRACT

Most of the existing music recommendation systems use collaborative or content based recommendation engines. However, the music choice of a user is not only dependent to the historical preferences or music contents. But also dependent to the mood of that user. This paper proposes an emotion based music recommendation framework that learns the emotion of a user from the signals obtained via wearable physiological sensors. In particular, the emotion of a user is classified by a wearable computing device which is integrated with a galvanic skin response (GSR) and photo plethysmography (PPG) physiological sensors. This emotion information is feed to any collaborative or content based recommendation engine as a supplementary data. Thus, existing recommendation engine performances can be increased using these data. Therefore, in this paper emotion recognition problem is considered as arousal and valence prediction from multi-channel physiological signals. Experimental results are obtained on 32 subjects' GSR and PPG signal data with/out feature fusion using decision tree, random forest, support vector machine and k-nearest neighbors algorithms. The results of comprehensive experiments on real data confirm the accuracy of the proposed emotion classification system that can be integrated to any recommendation engine. Index Terms—Emotion Aware Recommendation Engine, Emotion Recognition, Galvanic Skin Response, Machine Learning, Physiological Signals, Photo Plethysmography

CONTENTS

| S.No. | Chapters | Page No. |
|---------------------------|------------------------------|-----------------|
| i. | List of Figures..... | i |
| ii. | List of Screenshots..... | ii |
| 1. INTRODUCTION | | |
| 1.1 | INTRODUCTION OF PROJECT..... | 1 |
| 1.2 | LITERATURE SURVEY..... | 2 |
| 1.3 | MODULES..... | 4 |
| 2. SYSTEM ANALYSIS | | |
| 2.1 | EXISTING SYSTEM..... | 6 |
| 2.1.2 | DISADVANTAGES..... | 6 |
| 2.2 | PROPOSED SYSTEM | 7 |
| 2.2.1 | ADVANTAGES..... | 7 |
| 2.3 | SYSTEM REQUIREMENTS..... | 8 |
| 3. SYSTEM STUDY | | |
| 3.1 | FEASIBILITY STUDY..... | 9 |
| 3.1.1 | TECHNICAL FEASIBILITY..... | 10 |
| 3.1.2 | OPERATIONAL FEASIBILITY..... | 10 |
| 3.1.3 | ECONOMIC FEASIBILITY..... | 13 |
| 4. SYSTEM DESIGN | | |
| 4.1 | ARCHITECTURE..... | 14 |
| 4.2 | UML DIAGRAMS..... | 15 |
| 4.2.1 | USE CASE DIAGRAM..... | 16 |
| 4.2.2 | ACTIVITY DIAGRAM..... | 17 |
| 4.2.3 | CLASS DIAGRAM..... | 18 |

| | |
|---|-----------|
| 4.2.5 SEQUENCE DIAGRAM..... | 19 |
| 5.TECHNOLOGIES USED | |
| 5.1 WHAT IS PYTHON?..... | 20 |
| 5.1.1 ADVANTAGES & DISADVANTAGES OF PYTHON..... | 20 |
| 5.1.2 HISTORY..... | 24 |
| 5.2 WHAT IS MACHINE LEARNING ?..... | 25 |
| 5.2.1 CATEGORIES OF ML | 25 |
| 5.2.2 NEED FOR ML..... | 26 |
| 5.2.3 CHALLENGES IN ML..... | 26 |
| 5.2.4 APPLICATIONS..... | 27 |
| 5.2.5 HOW TO START LEARNING ML?..... | 28 |
| 5.2.6 ADVANTAGES & DISADVANTAGES OF ML..... | 31 |
| 5.3 PYTHON DEVELOPMENT STEPS..... | 32 |
| 5.4 MODULES USED IN PYTHON..... | 34 |
| 5.5 INSTALL PYTHON STEP BY STEP IN WINDOWS & MAC..... | 36 |
| 6. IMPLEMENTATION | |
| 6.1 SOFTWARE ENVIRONMENT..... | 44 |
| 6.1.1 PYTHON..... | 45 |
| 6.1.2 SAMPLE CODE..... | 46 |
| 7.SYSTEM TESTING | |
| 7.1 INTRODUCTION TO TESTING..... | 54 |
| 7.2 TESTING STRATEGIES..... | 55 |
| 8. SCREENSHOTS..... | 62 |
| 9. CONCLUSION..... | 66 |
| 10. REFERENCES..... | 67 |

LIST OF FIGURES

| Fig No | Name | Page No |
|---------------|------------------------|----------------|
| Fig.1 | Architecture diagram | 14 |
| Fig.2 | Use case diagram | 16 |
| Fig.3 | Activity diagram | 17 |
| Fig.4 | Class diagram | 18 |
| Fig.5 | Sequence diagram | 19 |
| Fig.6 | Installation of Python | 40 |

LIST OF SCREENSHOTS

| Fig No | Name | Page No |
|---------------|------------------------|----------------|
| Fig.1 | Upload Image With Face | 62 |
| Fig.2 | Select Image | 62 |
| Fig.3 | Pre-process | 63 |
| Fig.4 | Detect Face In Image | 63 |
| Fig.5 | Detect Emotion | 64 |
| Fig.6 | Play Song | 65 |

1.INTRODUCTION

1.1 INTRODUCTION TO PROJECT:

WEARABLE computing is the study or practice of inventing, designing, building or using body-worn computational and sensory devices that leverages a new type of human-computer interaction with a body-attached component that is always up and running. As the number of wearable computing device users are growing every year, their areas of utilization are also rapidly increasing. They have influenced medical care, fitness, aging, disabilities, education, transportation, finance, gaming, and music industries [1], [2] . Recommendation engines are algorithms which aim to provide the most relevant items to the user by filtering useful information from a huge pool of data. Recommendation engines may discover data patterns in the data set by learning user's choices and produce the outcomes that co-relates to their needs and interests [3]. Most of the recommender systems do not consider human emotions or expressions. However, emotions have noticeable influence on daily life of people. For a rich set of applications including human-robot interaction, computer aided tutoring, emotion aware interactive games, neuro marketing, socially intelligent software apps, computers Manuscript received March 31, 2018; revised May 26, 2018. The authors are with Department of Computer Engineering, Faculty of Computer and Informatics, Istanbul Technical University, Istanbul, Turkey (email:{ayatadeger, yyaslan, kamasak}@itu.edu.tr). should consider the emotions of their human conversation partners. Speech analytics and facial expressions [4], [5] have been used for emotion detection. However, in case of human beings prefer to camouflage their expressions, using only speech signals or facial expression signals may not be enough to detect emotions reliably. Compared with facial expressions, using physiological signals is a more reliable method to track and recognize emotions and internal cognitive processes of people. Our motivation in this work is to use emotion recognition techniques with wearable computing devices to generate additional inputs for music recommender system's algorithm, and to enhance the accuracy of the resulting music recommendations. In our previous works, we have studied emotion recognition from only GSR signals. In this study we are enriching signals with PPG and propose a data fusion based emotion recognition method for music recommendation engines [6]. The proposed wearable attached music recommendation framework utilizes not only the user's demographics but also his/her emotion state at the time of recommendation. Using GSR and PPG signals we have obtained promising results for emotion prediction.

1.2 LITERATURE SURVEY

TITLE: A METHODOLOGY TO DETECT TEMPORAL REGULARITIES IN USER BEHAVIOR FOR ANOMALY DETECTION.

AUTHORS: Alexandr Seleznyov, Finland.

ABSTRACT: Network security, and intrusion detection in particular, represents an area of increased interest in security community over last several years. However, the majority of work in this area has been concentrated upon implementation of misuse detection systems for intrusion patterns monitoring among network traffic. In anomaly detection the classification was mainly based on statistical or sequential analysis of data often neglecting temporal events' information as well as existing relations between them. In this paper we consider an anomaly detection problem as one of classification of user behavior in terms of incoming multiple discrete sequences. We present an approach that allows creating and maintaining user behavior profiles relying not only on sequential information but taking into account temporal features, such as events' lengths and possible relations between them. We define a user profile as a number of predefined classes of actions with accumulated temporal statistics for every class, and matrix of possible relations between classes.

TITLE: TARGETVUE: VISUAL ANALYSIS OF ANOMALOUS USER BEHAVIORS IN ONLINE COMMUNICATION SYSTEMS.

AUTHOR: Nan Cao, Conglei Shi, Sabrina Lin, Jie Lu, Yu-Ru Lin, Ching-Yung Lin

ABSTRACT: Users with anomalous behaviors in online communication systems (e.g. email and social medial platforms) are potential threats to society. Automated anomaly detection based on advanced machine learning techniques has been developed to combat this issue; challenges remain, though, due to the difficulty of obtaining proper ground truth for model training and evaluation. Therefore, substantial human judgment on the automated analysis results is often required to better adjust the performance of anomaly detection. Unfortunately, techniques that allow users to understand the analysis results more efficiently, to make a confident judgment about anomalies, and to explore data in their context, are still lacking. In this paper, we propose a novel visual analysis system, TargetVue, which detects anomalous users via an unsupervised learning model and visualizes the behaviors of suspicious users in behavior-rich context through novel visualization designs and multiple coordinated contextual views. Particularly, TargetVue incorporates three new ego-centric glyphs to visually

summarize a user's behaviors which effectively present the user's communication activities, features, and social interactions. An efficient layout method is proposed to place these glyphs on a triangle grid, which captures similarities among users and facilitates comparisons of behaviors of different users. We demonstrate the power of TargetVue through its application in a social bot detection challenge using Twitter data, a case study based on email records, and an interview with expert users. Our evaluation shows that TargetVue is beneficial to the detection of users with anomalous communication behavior.

TITLE: MINING SOCIAL NETWORKS FOR ANOMALIES: METHODS AND CHALLENGES

AUTHOR: P V Bindua, P Santhi Thilagama, India.

ABSTRACT: Online social networks have received a dramatic increase of interest in the last decade due to the growth of Internet and Web 2.0. They are among the most popular sites on the Internet that are being used in almost all areas of life including education, medical, entertainment, business, and telemarketing. Unfortunately, they have become primary targets for malicious users who attempt to perform illegal activities and cause harm to other users. The unusual behavior of such users can be identified by using anomaly detection techniques. Anomaly detection in social networks refers to the problem of identifying the strange and unexpected behavior of users by exploring the patterns hidden in the networks, as the patterns of interaction of such users deviate significantly from the normal users of the networks. Even though a multitude of anomaly detection methods have been developed for different problem settings, this field is still relatively young and rapidly growing. Hence, there is a growing need for an organized study of the work done in the area of anomaly detection in social networks. In this paper, we provide a comprehensive review of a large set of methods for mining social networks for anomalies by providing a multi-level taxonomy to categorize the existing techniques based on the nature of input network, the type of anomalies they detect, and the underlying anomaly detection approach. In addition, this paper highlights the various application scenarios where these methods have been used, and explores the research challenges and open issues in this field.

1.3 MODULES

- Data Collection and Processing
- Recommender Model
- Recommendation Post-processing
- Feedback
- User Interface

A. Data collection and Processing

The Data Collection and Processing module in an Emotion-Based Music Responsive System is a crucial component responsible for gathering, analyzing, and interpreting data related to user emotions to generate appropriate music responses. This module integrates various technologies and techniques to ensure accurate emotion detection and real-time music adaptation, creating a seamless and immersive user experience.

B. Recommender Model

The Recommender Model module in an Emotion-Based Music Responsive System is a key component responsible for leveraging data analysis and machine learning techniques to provide personalized music recommendations tailored to the user's emotional state. This module employs various algorithms and strategies to understand user preferences, analyze emotional context, and deliver music content that aligns with the user's mood and preferences.

C. Recommendation Post-processing

The Recommendation Post-processing module in an Emotion-Based Music Responsive System plays a crucial role in refining and enhancing the music recommendations generated by the Recommender Model. This module employs additional processing steps and algorithms to ensure that the recommended music content aligns well with the user's emotional state, preferences, and overall listening experience. The post-processing stage focuses on improving recommendation quality, diversity, relevance, and user satisfaction.

D. Feedback

The Feedback Module in an Emotion-Based Music Responsive System is a vital component that facilitates interaction between the system and the user, allowing for feedback collection, analysis, and integration to improve the overall user experience. This module plays a crucial role in gathering insights into user preferences, satisfaction levels, emotional responses, and music preferences, which in turn helps in refining the system's recommendations and responsiveness.

E. User Interface

The User Interface (UI) module in an Emotion-Based Music Responsive System is responsible for providing a user-friendly and intuitive interface through which users can interact with the system, control music playback, view recommendations, and provide feedback. This module plays a crucial role in enhancing user experience, ensuring ease of use, and facilitating seamless communication between the user and the system.

2. SYSTEM ANALYSIS

2.1 EXISTING SYSTEM:

Coefficient of reliability is calculated at each step of classification. During classification a coefficient of reliability is changed. Based on this, normal or anomalous user behavior is identified. While classifying the user behavior the system monitors deviations between expected user behavior and current one. Coefficient of reliability is used to estimate the value of deviation in user behavior. If this value crosses a certain threshold, then it is considered as a case of abnormal behavior. That means the parameters in actions of user are not in admissible intervals. Coefficient of reliability is measuring the same individual twice and it correlates the 2 sets of measures. Every user action class was characterized by statistic parameters of time distribution – mean and standard deviation. Deviations from current values of sequential and temporal parameters are considered as consequence of abnormal behavior. The tools which are used to classify user behavior are N action classes and a relational matrix. These tools describe the model of user behavior.

DISADVANTAGES:

- Users who don't have internet connection can't access the system.
- System does not display products to the user which are purchased frequently.

2.2 PROPOSED SYSTEM:

In this work, we propose to overcome the limitations of prior works in user preference modeling by exploring local and global user behavior patterns on a user successive behavior graph (SBG), which is constructed by utilizing short-term actions of all users. We then exploit high-order relations in the SBG to capture implicit collaborative patterns and preference signals with an efficient jumping graph convolution and learn enriched product representations for user preference modeling. Our approach addresses the aforementioned problems in the following two aspects.

ADVANTAGES:

- This system helps to find out products which are more in demand.
- This system provides the data in graphical format.
- As user behavior pattern is put up in graphical format it will be easier for the admin to view the data and can make decision process faster and can come up with solution quicker.
- This system helps the admin to know most frequently purchased products by the customer.

2.3 SYSTEM REQUIREMENTS:

HARDWARE REQUIREMENTS:

| | |
|------------------|----------------------|
| Processor | i3 processor 5th Gen |
| RAM | 4GB |
| Hard Disk | 500 GB |

SOFTWARE REQUIREMENTS:

| | |
|--|---------------------------------------|
| Operating System | Windows 10/11 |
| Programming Language | Python 3.10 |
| Domain | Image Processing & Cloud Computing |
| Integrated Development Environment (IDE) | Visual Studio Code |
| Front End Technologies | HTML5, CSS3, Java Script |
| Back End Technologies or Framework | Django |
| Database(RDBMS) | MySQL |
| Database software | WAMP or XAMPP server |
| Web Server or Deployment Server | Django Application Development Server |
| Design/Modelling | Rational Rose |

3. SYSTEM STUDY

3.1 FEASIBILITY STUDY

1. TECHNICAL FEASIBILITY
2. OPERATIONAL FEASIBILITY
3. ECONOMIC FEASIBILITY

INTRODUCTION

A feasibility study assesses the operational, technical and economic merits of the proposed project. The feasibility study is intended to be a preliminary review of the facts to see if it is worthy of proceeding to the analysis phase. From the systems analyst perspective, the feasibility analysis is the primary tool for recommending whether to proceed to the next phase or to discontinue the project.

The feasibility study is a management-oriented activity. The objective of a feasibility study is to find out if an information system project can be done and to suggest possible alternative solutions.

Projects are initiated for two broad reasons:

1. Problems that lend themselves to systems solutions
2. Opportunities for improving through:
 - (a) upgrading systems
 - (b) altering systems
 - (c) installing new systems

A feasibility study should provide management with enough information to decide:

- Whether the project can be done
- Whether the final product will benefit its intended users and organization
- What are the alternatives among which a solution will be chosen
- Is there a preferred alternative?

3.1.1 TECHNICAL FEASIBILITY

A large part of determining resources has to do with assessing technical feasibility. It considers the technical requirements of the proposed project. The technical requirements are then compared to the technical capability of the organization. The systems project is considered technically feasible if the internal technical capability is sufficient to support the project requirements.

The analyst must find out whether current technical resources can be upgraded or added to in a manner that fulfills the request under consideration. This is where the expertise of system analysts is beneficial, since using their own experience and their contact with vendors they will be able to answer the question of technical feasibility.

The essential questions that help in testing the operational feasibility of a system include the following:

- Is the project feasible within the limits of current technology?
- Does the technology exist at all?
- Is it available within given resource constraints?
- Is it a practical proposition?
- Manpower- programmers, testers & debuggers
- Software and hardware
- Are the current technical resources sufficient for the new system?
- Can they be upgraded to provide the level of technology necessary for the new system?
- Do we possess the necessary technical expertise, and is the schedule reasonable?
- Can the technology be easily applied to current problems?
- Does the technology have the capacity to handle the solution?
- Do we currently possess the necessary technology?

3.1.2 OPERATIONAL FEASIBILITY

Operational feasibility is dependent on human resources available for the project and involves projecting whether the system will be used if it is developed and implemented. Operational feasibility is a measure of how well a proposed system solves the problems, and takes advantage of the opportunities identified during scope definition and how it satisfies the requirements identified in the requirements analysis phase of system development.

Operational feasibility reviews the willingness of the organization to support the proposed system. This is probably the most difficult of the feasibilities to gauge. In order to determine this feasibility, it is important to understand the management commitment to the proposed project. If the request was initiated by management, it is likely that there is management support and the system will be accepted and used. However, it is also important that the employee base will be accepting of the change.

The essential questions that help in testing the operational feasibility of a system include the following:

- Does current mode of operation provide adequate throughput and response time?
- Does current mode provide end users and managers with timely, pertinent, accurate and useful formatted information?
- Does current mode of operation provide cost-effective information services to the business?
- Could there be a reduction in cost and or an increase in benefits?
- Does current mode of operation offer effective controls to protect against fraud and to guarantee accuracy and security of data and information?
- Does current mode of operation make maximum use of available resources, including people, time, and flow of forms?
- Does current mode of operation provide reliable services
- Are the services flexible and expandable?
- Are the current work practices and procedures adequate to support the new system?
- If the system is developed, will it be used?
- Manpower problems
- Labour objections
- Manager resistance
- Organizational conflicts and policies
- Social acceptability
- Government regulations
- Does management support the project?
- Are the users not happy with current business practices?
- Will it reduce the time (operation) considerably?
- Have the users been involved in the planning and development of the project?
- Will the proposed system really benefit the organization?
- Does the overall response increase?

- Will accessibility of information be lost?
- Will the system affect the customers in considerable way?
- Legal aspects
- How do the end-users feel about their role in the new system?
- What end-users or managers may resist or not use the system?
- How will the working environment of the end-user change?
- Can or will end-users and management adapt to the change?

3.1.3 ECONOMIC FEASIBILITY

Economic analysis could also be referred to as cost/benefit analysis. It is the most frequently used method for evaluating the effectiveness of a new system. In economic analysis the procedure is to determine the benefits and savings that are expected from a candidate system and compare them with costs. If benefits outweigh costs, then the decision is made to design and implement the system. An entrepreneur must accurately weigh the cost versus benefits before taking an action.

Possible questions raised in economic analysis are:

- Is the system cost effective?
- Do benefits outweigh costs?
- The cost of doing full system study
- The cost of business employee time
- Estimated cost of hardware
- Estimated cost of software/software development
- Is the project possible, given the resource constraints?
- What are the savings that will result from the system?
- Cost of employees' time for study
- Cost of packaged software/software development
- Selection among alternative financing arrangements (rent/lease/purchase)

The concerned business must be able to see the value of the investment it is pondering before committing to an entire system study. If short-term costs are not overshadowed by long-term gains or produce no immediate reduction in operating costs, then the system is not economically feasible, and the project should not proceed any further. If the expected benefits

equal or exceed costs, the system can be judged to be economically feasible. Economic analysis is used for evaluating the effectiveness of the proposed system.

The economic feasibility will review the expected costs to see if they are in-line with the projected budget or if the project has an acceptable return on investment. At this point, the projected costs will only be a rough estimate. The exact costs are not required to determine economic feasibility. It is only required to determine if it is feasible that the project costs will fall within the target budget or return on investment. A rough estimate of the project schedule is required to determine if it would be feasible to complete the systems project within a required timeframe. The required timeframe would need to be set by the organization.

4. SYSTEM DESIGN

4.1 SYSTEM ARCHITECTURE:

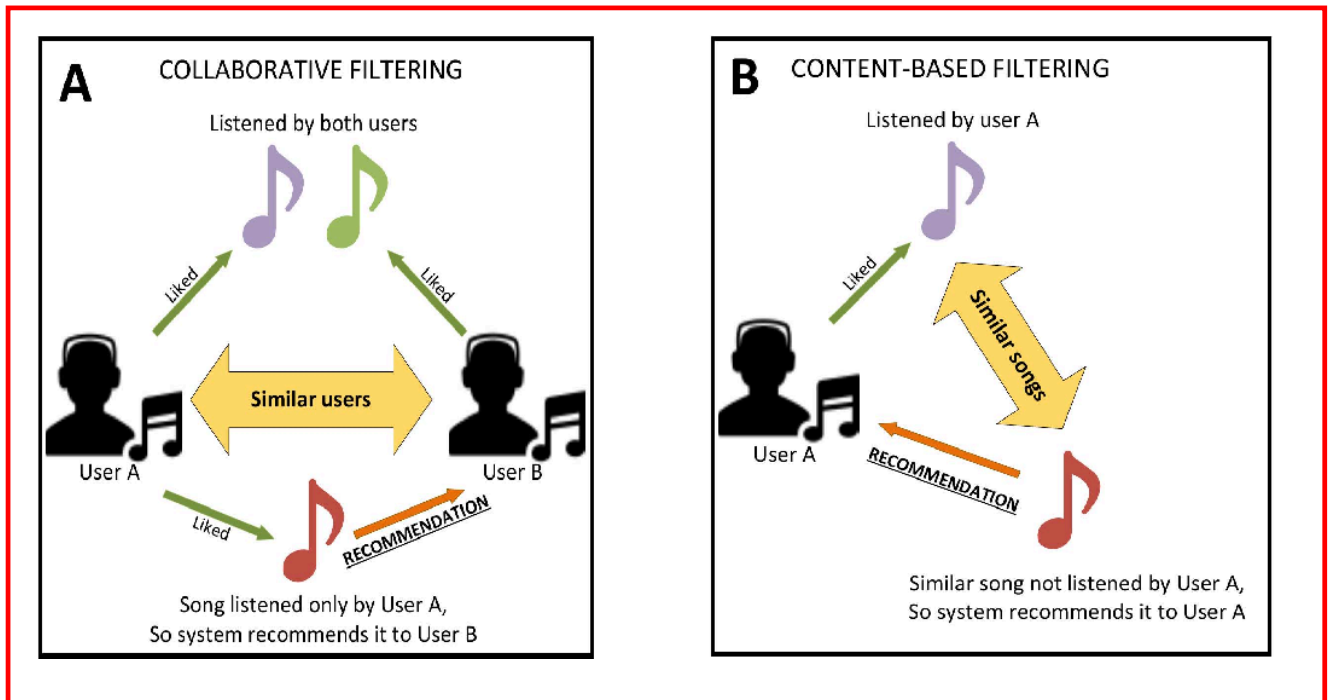


Fig. 1

4.2 UML DIAGRAMS

UML stands for Unified Modeling Language. UML is a standardized general-purpose modeling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group. The goal is for UML to become a common language for creating models of object oriented computer software. In its current form UML is comprised of two major components: a Meta-model and a notation. In the future, some form of method or process may also be added to or associated with, UML. The Unified Modeling Language is a standard language for specifying, Visualization, Constructing and documenting the artifacts of software system, as well as for business modeling and other nonsoftware systems. The UML represents a collection of best engineering practices that have proven successful in the modeling of large and complex systems. The UML is a very important part of developing objects oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects.

GOALS:

The Primary goals in the design of the UML are as follows:

1. Provide users a ready-to-use, expressive visual modeling Language so that they can develop and exchange meaningful models.
2. Provide extendibility and specialization mechanisms to extend the core concepts.
3. Be independent of particular programming languages and development process.
4. Provide a formal basis for understanding the modeling language.
5. Encourage the growth of OO tools market.
6. Support higher level development concepts such as collaborations, frameworks, patterns and components.
7. Integrate best practices.

4.2.1 USE CASE DIAGRAM:

A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.

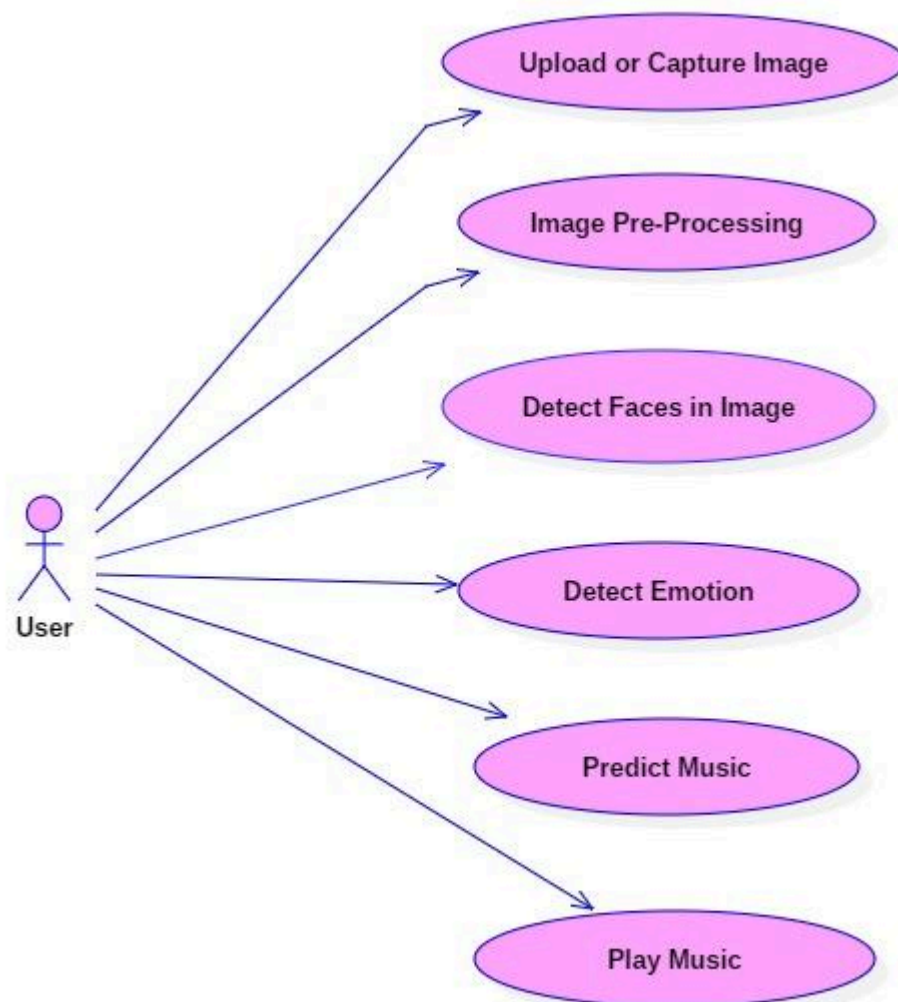


Fig. 2

4.2.2 ACTIVITY DIAGRAM:

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modeling Language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control.

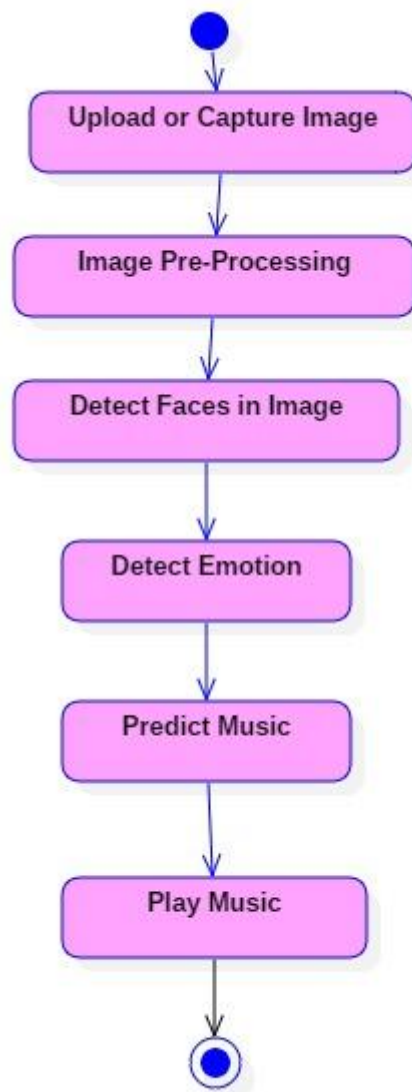


Fig. 3

4.2.3 CLASS DIAGRAM

Class diagram is a static diagram. It represents the static view of an application. Class diagram is not only used for visualizing, describing, and documenting different aspects of a system but also for constructing executable code of the software application. Class diagram describes the attributes and operations of a class and also the constraints imposed on the system. The class diagrams are widely used in the modeling of object oriented systems because they are the only UML diagrams, which can be mapped directly with object-oriented languages.

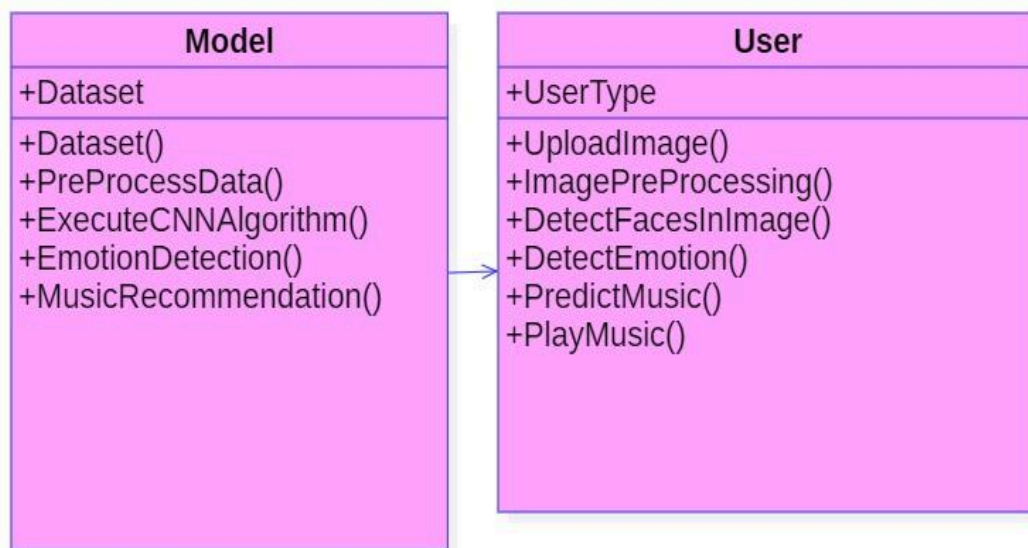


Fig. 4

4.2.5 SEQUENCE DIAGRAM:

A sequence diagram in Unified Modeling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. Sequence diagrams are sometimes called event diagrams, event scenarios, and timing diagrams.

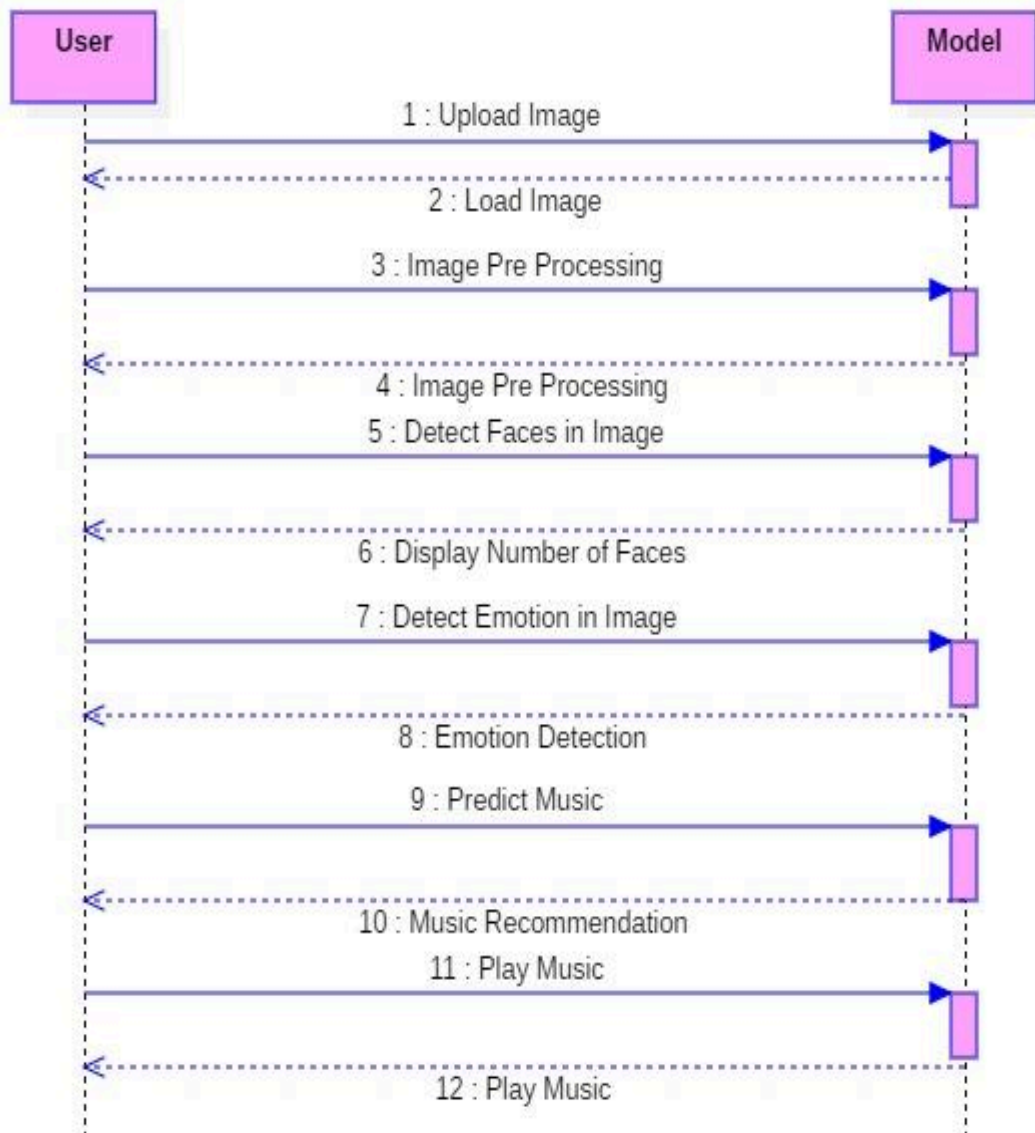


Fig. 5

5.TECHNOLOGIES

5.1 WHAT IS PYTHON?

Below are some facts about Python. Python is currently the most widely used multi-purpose, high-level programming language. Python allows programming in Object-Oriented and Procedural paradigms. Python programs generally are smaller than other programming languages like Java. Programmers have to type relatively less and indentation requirement of the language, makes them readable all the time. Python language is being used by almost all tech-giant companies like – Google, Amazon, Facebook, Instagram, Dropbox, Uber... etc.

The biggest strength of Python is huge collection of standard library which can be used for the following –

- Machine Learning
- GUI Applications (like Kivy, Tkinter, PyQt etc.)
- Web frameworks like Django (used by YouTube, Instagram, Dropbox)
- Image processing (like Opencv, Pillow)
- Web scraping (like Scrapy, BeautifulSoup, Selenium)
- Test frameworks
- Multimedia

5.1.1 ADVANTAGES & DISADVANTAGES OF PYTHON

Advantages of Python :-

Let's see how Python dominates over other languages.

1.Extensive Libraries

Python downloads with an extensive library and it contain code for various purposes like regular expressions, documentation-generation, unit-testing, web browsers, threading, databases, CGI, email, image manipulation, and more. So, we don't have to write the complete code for that manually.

2. Extensible

As we have seen earlier, Python can be **extended to other languages**. You can write some of your code in languages like C++ or C. This comes in handy, especially in projects.

3. Embeddable

Complimentary to extensibility, Python is embeddable as well. You can put your Python code in your source code of a different language, like C++. This lets us add **scripting capabilities** to our code in the other language.

4. Improved Productivity

The language's need to be in simplicity and extensive libraries render programmers **more productive** than languages like Java and C++ do. Also, the fact that you need to write less and get more things done.

5. IOT Opportunities

Since Python forms the basis of new platforms like Raspberry Pi, it finds the future bright for the Internet Of Things. This is a way to connect the language with the real world.

6. Simple and Easy

When working with Java, you may have to create a class to print '**Hello World**'. But in Python, just a print statement will do. It is also quite **easy to learn, understand, and code**. This is why when people pick up Python, they have a hard time adjusting to other more verbose languages like Java.

7. Readable

Because it is not such a verbose language, reading Python is much like reading English. This is the reason why it is so easy to learn, understand, and code. It also does not need curly braces to define blocks, and **indentation is mandatory**. This further aids the readability of the code.

8. Object-Oriented

This language supports both the **procedural and object-oriented** programming paradigms. While functions help us with code reusability, classes and objects let us model the real world. A class allows the **encapsulation of data** and functions into one.

9. Free and Open-Source

Like we said earlier, Python is **freely available**. But not only can you **download Python** for free, but you can also download its source code, make changes to it, and even distribute it. It downloads with an extensive collection of libraries to help you with your tasks.

10. Portable

When you code your project in a language like C++, you may need to make some changes to it if you want to run it on another platform. But it isn't the same with Python. Here, you need to **code only once**, and you can run it anywhere. This is called **Write Once Run Anywhere (WORA)**. However, you need to be careful enough not to include any system dependent features.

11. Interpreted

Lastly, we will say that it is an interpreted language. Since statements are executed one by one, **debugging is easier** than in compiled languages.

Advantages of Python Over Other Languages

1. Less Coding

Almost all of the tasks done in Python requires less coding when the same task is done in other languages. Python also has an awesome standard library support, so you don't have to search for any third-party libraries to get your job done. This is the reason that many people suggest learning Python to beginners.

2. Affordable

Python is free therefore individuals, small companies or big organizations can leverage the free available resources to build applications. Python is popular and widely used so it gives you better community support.

The 2019 Github annual survey showed us that Python has overtaken Java in the most popular programming language category.

3. Python is for Everyone

Python code can run on any machine whether it is Linux, Mac or Windows. Programmers need to learn different languages for different jobs but with Python, you can professionally build web apps, perform data analysis and **machine learning**, automate things, do web scraping and also build games and powerful visualizations. It is an all-rounder programming language.

Disadvantages of Python

So far, we've seen why Python is a great choice for your project. But if you choose it, you should be aware of its consequences as well. Let's now see the downsides of choosing Python over another language.

1. Speed Limitations

We have seen that Python code is executed line by line. But since Python is interpreted, it often results in **slow execution**. This, however, isn't a problem unless speed is a focal point for the project. In other words, unless high speed is a requirement, the benefits offered by Python are enough to distract us from its speed limitations.

2. Weak in Mobile Computing and Browsers

While it serves as an excellent server-side language, Python is much rarely seen on the **client-side**. Besides that, it is rarely ever used to implement smartphone-based applications. One such application is called **Carbonnelle**.

The reason it is not so famous despite the existence of Brython is that it isn't that secure.

3. Design Restrictions

As you know, Python is **dynamically-typed**. This means that you don't need to declare the type of variable while writing the code. It uses **duck-typing**. But wait, what's that? Well, it just means that if it looks like a duck, it must be a duck. While this is easy on the programmers during coding, it can **raise run-time errors**.

4. Underdeveloped Database Access Layers

Compared to more widely used technologies like **JDBC (Java DataBase Connectivity)** and **ODBC (Open DataBase Connectivity)**, Python's database access layers are a bit underdeveloped. Consequently, it is less often applied in huge enterprises.

5. Simple

No, we're not kidding. Python's simplicity can indeed be a problem. Take my example. I don't do Java, I'm more of a Python person. To me, its syntax is so simple that the verbosity of Java code seems unnecessary.

This was all about the Advantages and Disadvantages of Python Programming Language.

5.1.2 HISTORY OF PYTHON

What do the alphabet and the programming language Python have in common? Right, both start with ABC. If we are talking about ABC in the Python context, it's clear that the programming language ABC is meant. ABC is a general-purpose programming language and programming environment, which had been developed in the Netherlands, Amsterdam, at the CWI (Centrum Wiskunde & Informatica). The greatest achievement of ABC was to influence the design of Python. Python was conceptualized in the late 1980s. Guido van Rossum worked that time in a project at the CWI, called Amoeba, a distributed operating system. In an interview with Bill Venners¹, Guido van Rossum said: "In the early 1980s, I worked as an implementer on a team building a language called ABC at Centrum Wiskunde en Informatica (CWI). I don't know how well people know ABC's influence on Python. I try to mention ABC's influence because I'm indebted to everything I learned during that project and to the people who worked on it. Later on in the same Interview, Guido van Rossum continued: "I remembered all my experience and some of my frustration with

ABC. I decided to try to design a simple scripting language that possessed some of ABC's better properties, but without its problems. So I started typing. I created a simple virtual machine, a simple parser, and a simple runtime. I made my own version of the various ABC parts that I liked. I created a basic syntax, used indentation for statement grouping instead of curly braces or begin-end blocks, and developed a small number of powerful data types: a hash table (or dictionary, as we call it), a list, strings, and numbers."

5.2 WHAT IS MACHINE LEARNING?

Before we take a look at the details of various machine learning methods, let's start by looking at what machine learning is, and what it isn't. Machine learning is often categorized as a subfield of artificial intelligence, but I find that categorization can often be misleading at first brush. The study of machine learning certainly arose from research in this context, but in the data science application of machine learning methods, it's more helpful to think of machine learning as a means of building models of data.

Fundamentally, machine learning involves building mathematical models to help understand data. "Learning" enters the fray when we give these models *tunable parameters* that can be adapted to observed data; in this way the program can be considered to be "learning" from the data. Once these models have been fit to previously seen data, they can be used to predict and understand aspects of newly observed data. I'll leave to the reader the more philosophical digression regarding the extent to which this type of mathematical, model-based "learning" is similar to the "learning" exhibited by the human brain. Understanding the problem setting in machine learning is essential to using these tools effectively, and so we will start with some broad categorizations of the types of approaches we'll discuss here.

5.2.1 Categories Of Machine Learning

At the most fundamental level, machine learning can be categorized into two main types: supervised learning and unsupervised learning.

Supervised learning involves somehow modeling the relationship between measured features of data and some label associated with the data; once this model is determined, it can be used to apply labels to new, unknown data. This is further subdivided into

classification tasks and regression tasks: in classification, the labels are discrete categories, while in regression, the labels are continuous quantities. We will see examples of both types of supervised learning in the following section.

Unsupervised learning involves modeling the features of a dataset without reference to any label, and is often described as "letting the dataset speak for itself." These models include tasks such as *clustering* and *dimensionality reduction*. Clustering algorithms identify distinct groups of data, while dimensionality reduction algorithms search for more succinct representations of the data. We will see examples of both types of unsupervised learning in the following section.

5.2.2 Need for Machine Learning

Human beings, at this moment, are the most intelligent and advanced species on earth because they can think, evaluate and solve complex problems. On the other side, AI is still in its initial stage and haven't surpassed human intelligence in many aspects. Then the question is that what is the need to make machine learn? The most suitable reason for doing this is, "to make decisions, based on data, with efficiency and scale".

Lately, organizations are investing heavily in newer technologies like Artificial Intelligence, Machine Learning and Deep Learning to get the key information from data to perform several real-world tasks and solve problems. We can call it data-driven decisions taken by machines, particularly to automate the process. These data-driven decisions can be used, instead of using programming logic, in the problems that cannot be programmed inherently. The fact is that we can't do without human intelligence, but other aspect is that we all need to solve real-world problems with efficiency at a huge scale. That is why the need for machine learning arises.

5.2.3 Challenges in Machines Learning

While Machine Learning is rapidly evolving, making significant strides with cybersecurity and autonomous cars, this segment of AI as whole still has a long way to go. The reason behind is that ML has not been able to overcome number of challenges. The challenges that ML is facing currently are –

Quality of data – Having good-quality data for ML algorithms is one of the biggest challenges. Use of low-quality data leads to the problems related to data preprocessing and feature extraction.

Time-Consuming task – Another challenge faced by ML models is the consumption of time especially for data acquisition, feature extraction and retrieval.

Lack of specialist persons – As ML technology is still in its infancy stage, availability of expert resources is a tough job.

No clear objective for formulating business problems – Having no clear objective and well-defined goal for business problems is another key challenge for ML because this technology is not that mature yet.

Issue of overfitting & underfitting – If the model is overfitting or underfitting, it cannot be represented well for the problem.

Curse of dimensionality – Another challenge ML model faces is too many features of data points. This can be a real hindrance.

Difficulty in deployment – Complexity of the ML model makes it quite difficult to be deployed in real life.

5.2.4 Applications of Machines Learning :-

Machine Learning is the most rapidly growing technology and according to researchers we are in the golden year of AI and ML. It is used to solve many real-world complex problems which cannot be solved with traditional approach. Following are some real-world applications of ML

—

- Emotion analysis
- Sentiment analysis
- Error detection and prevention
- Weather forecasting and prediction
- Stock market analysis and forecasting
- Speech synthesis
- Speech recognition
- Customer segmentation

- Object recognition
- Fraud detection
- Fraud prevention
- Recommendation of products to customer in online shopping

5.2.5 How to Start Learning Machine Learning?

Arthur Samuel coined the term “**Machine Learning**” in 1959 and defined it as a “**Field of study that gives computers the capability to learn without being explicitly programmed**”.

And that was the beginning of Machine Learning! In modern times, Machine Learning is one of the most popular (if not the most!) career choices. According to Indeed, Machine Learning Engineer Is The Best Job of 2019 with a 344% growth and an average base salary of **\$146,085** per year.

But there is still a lot of doubt about what exactly is Machine Learning and how to start learning it? So this article deals with the Basics of Machine Learning and also the path you can follow to eventually become a full-fledged Machine Learning Engineer. Now let's get started!!!

How to start learning ML?

This is a rough roadmap you can follow on your way to becoming an insanely talented Machine Learning Engineer. Of course, you can always modify the steps according to your needs to reach your desired end-goal!

Step 1 – Understand the Prerequisites

In the case, you are a genius, you could start ML directly but normally, there are some prerequisites that you need to know which include Linear Algebra, Multivariate Calculus, Statistics, and Python. And if you don't know these, never fear! You don't need Ph.D.degree in these topics to get started but you do need a basic understanding.

(a) Learn Linear Algebra and Multivariate Calculus

Both Linear Algebra and Multivariate Calculus are important in Machine Learning. However, the extent to which you need them depends on your role as a data scientist. If you

are more focused on application heavy machine learning, then you will not be that heavily focused on maths as there are many common libraries available. But if you want to focus on R&D in Machine Learning, then mastery of Linear Algebra and Multivariate Calculus is very important as you will have to implement many ML algorithms from scratch.

(b) Learn Statistics

Data plays a huge role in Machine Learning. In fact, around 80% of your time as an ML expert will be spent collecting and cleaning data. And statistics is a field that handles the collection, analysis, and presentation of data. So it is no surprise that you need to learn it!!! Some of the key concepts in statistics that are important are Statistical Significance, Probability Distributions, Hypothesis Testing, Regression, etc. Also, Bayesian Thinking is also a very important part of ML which deals with various concepts like Conditional Probability, Priors, and Posteriors, Maximum Likelihood, etc.

(c) Learn Python

Some people prefer to skip Linear Algebra, Multivariate Calculus and Statistics and learn them as they go along with trial and error. But the one thing that you absolutely cannot skip is Python! While there are other languages you can use for Machine Learning like R, Scala, etc. Python is currently the most popular language for ML. In fact, there are many Python libraries that are specifically useful for Artificial Intelligence and Machine Learning such as Keras, TensorFlow, Scikit-learn, etc. So if you want to learn ML, it's best if you learn Python! You can do that using various online resources and courses such as **Fork Python** available Free on GeeksforGeeks.

Step 2 – Learn Various ML Concepts

Now that you are done with the prerequisites, you can move on to actually learning ML (Which is the fun part!!!) It's best to start with the basics and then move on to more complicated stuff. Some of the basic concepts in ML are:

(a) Terminologies of Machine Learning

- **Model** – A model is a specific representation learned from data by applying some machine learning algorithm. A model is also called a hypothesis.
- **Feature** – A feature is an individual measurable property of the data. A set of numeric features can be conveniently described by a feature vector. Feature vectors are fed as input to the model. For example, in order to predict a fruit, there may be features like color, smell, taste, etc.
- **Target (Label)** – A target variable or label is the value to be predicted by our model. For the fruit example discussed in the feature section, the label with each set of input would be the name of the fruit like apple, orange, banana, etc.
- **Training** – The idea is to give a set of inputs(features) and it's expected outputs(labels), so after training, we will have a model (hypothesis) that will then map new data to one of the categories trained on.
- **Prediction** – Once our model is ready, it can be fed a set of inputs to which it will provide a predicted output(label).

(b) Types of Machine Learning

- **Supervised Learning** – This involves learning from a training dataset with labeled data using classification and regression models. This learning process continues until the required level of performance is achieved.
- **Unsupervised Learning** – This involves using unlabelled data and then finding the underlying structure in the data in order to learn more and more about the data itself using factor and cluster analysis models.
- **Semi-supervised Learning** – This involves using unlabelled data like Unsupervised Learning with a small amount of labeled data. Using labeled data vastly increases the learning accuracy and is also more cost-effective than Supervised Learning.
- **Reinforcement Learning** – This involves learning optimal actions through trial and error. So the next action is decided by learning behaviors that are based on the current state and that will maximize the reward in the future.

5.2.6 ADVANTAGES & DISADVANTAGES OF ML

Advantages of Machine learning :-

1. Easily identifies trends and patterns -

Machine Learning can review large volumes of data and discover specific trends and patterns that would not be apparent to humans. For instance, for an e-commerce website like Amazon, it serves to understand the browsing behaviors and purchase histories of its users to help cater to the right products, deals, and reminders relevant to them. It uses the results to reveal relevant advertisements to them.

2. No human intervention needed (automation)

With ML, you don't need to babysit your project every step of the way. Since it means giving machines the ability to learn, it lets them make predictions and also improve the algorithms on their own. A common example of this is anti-virus softwares. they learn to filter new threats as they are recognized. ML is also good at recognizing spam.

3. Continuous Improvement

As **ML algorithms** gain experience, they keep improving in accuracy and efficiency. This lets them make better decisions. Say you need to make a weather forecast model. As the amount of data you have keeps growing, your algorithms learn to make more accurate predictions faster.

4. Handling multi-dimensional and multi-variety data

Machine Learning algorithms are good at handling data that are multi-dimensional and multi variety, and they can do this in dynamic or uncertain environments.

5. Wide Applications

You could be an e-tailer or a healthcare provider and make ML work for you. Where it does apply, it holds the capability to help deliver a much more personal experience to customers while also targeting the right customers.

Disadvantages of Machine Learning :-

1. Data Acquisition

Machine Learning requires massive data sets to train on, and these should be inclusive/unbiased, and of good quality. There can also be times where they must wait for new data to be generated.

2. Time and Resources

ML needs enough time to let the algorithms learn and develop enough to fulfill their purpose with a considerable amount of accuracy and relevancy. It also needs massive resources to function. This can mean additional requirements of computer power for you.

3. Interpretation of Results

Another major challenge is the ability to accurately interpret results generated by the algorithms. You must also carefully choose the algorithms for your purpose.

4. High error-susceptibility

Machine Learning is autonomous but highly susceptible to errors. Suppose you train an algorithm with data sets small enough to not be inclusive. You end up with biased predictions coming from a biased training set. This leads to irrelevant advertisements being displayed to customers. In the case of ML, such blunders can set off a chain of errors that can go undetected for long periods of time. And when they do get noticed, it takes quite some time to recognize the source of the issue, and even longer to correct it.

5.3 PYTHON DEVELOPMENT STEPS

Guido Van Rossum published the first version of Python code (version 0.9.0) at alt.sources in February 1991. This release included already exception handling, functions, and the core data types of list, dict, str and others. It was also object oriented and had a module system. Python version 1.0 was released in January 1994. The major new features included in this release were the functional programming tools lambda, map, filter and reduce, which Guido Van Rossum never liked. Six and a half years later in October 2000, Python 2.0 was

introduced. This release included list comprehensions, a full garbage collector and it was supporting Unicode Python flourished for another 8 years in the versions 2.x before the next major release as Python 3.0 (also known as "Python 3000" and "Py3K") was released. Python 3 is not backwards compatible with Python 2.x. The emphasis in Python 3 had been on the removal of duplicate programming constructs and modules, thus fulfilling or coming close to fulfilling the 13th law of the Zen of Python: "There should be one -- and preferably only one -- obvious way to do it. Some changes in Python 7.3:

- Print is now a function
- Views and iterators instead of lists
- The rules for ordering comparisons have been simplified. E.g. a heterogeneous list cannot be sorted, because all the elements of a list must be comparable to each other.
- There is only one integer type left, i.e. int. long is int as well.
- The division of two integers returns a float instead of an integer. "/" can be used to have the "old" behaviour.
- Text Vs. Data Instead Of Unicode Vs. 8-bit

Purpose :-

We demonstrated that our approach enables successful segmentation of intra-retinal layers—even with low-quality images containing speckle noise, low contrast, and different intensity ranges throughout—with the assistance of the ANIS feature.

Python

Python is an interpreted high-level programming language for general-purpose programming. Created by Guido van Rossum and first released in 1991, Python has a design philosophy that emphasizes code readability, notably using significant whitespace.

Python features a dynamic type system and automatic memory management. It supports multiple programming paradigms, including object-oriented, imperative, functional and procedural, and has a large and comprehensive standard library.

- Python 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 also acknowledges that speed of development is important. Readable and terse code is part of this, and so is access to powerful constructs that avoid tedious repetition of code. Maintainability also ties into this may be an all but useless metric, but it does say something about how much code you have to scan, read and/or understand to troubleshoot problems or tweak behaviors. This speed of development, the ease with which a programmer of other languages can pick up basic Python skills and the huge standard library is key to another area where Python excels. All its tools have been quick to implement, saved a lot of time, and several of them have later been patched and updated by people with no Python background - without breaking.

5.4 MODULES USED IN PROJECT

Tensorflow

TensorFlow is a free and open-source software library for dataflow and differentiable programming across a range of tasks. It is a symbolic math library, and is also used for machine learning applications such as neural networks. It is used for both research and production at Google.

TensorFlow was developed by the Google Brain team for internal Google use. It was released under the Apache 2.0 open-source license on November 9, 2015.

Numpy

Numpy is a general-purpose array-processing package. It provides a high-performance multidimensional array object, and tools for working with these arrays.

It is the fundamental package for scientific computing with Python. It contains various features including these important ones:

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

Pandas

Pandas is an open-source Python Library providing high-performance data manipulation and analysis tool using its powerful data structures. Python was majorly used for data munging and preparation. It had very little contribution towards data analysis. Pandas solved this problem. Using Pandas, we can accomplish five typical steps in the processing and analysis of data, regardless of the origin of data load, prepare, manipulate, model, and analyze. Python with Pandas is used in a wide range of fields including academic and commercial domains including finance, economics, Statistics, analytics, etc.

Matplotlib

Matplotlib is a Python 2D plotting library which produces publication quality figures in a variety of hardcopy formats and interactive environments across platforms. Matplotlib can be used in Python scripts, the Python and IPython shells, the Jupyter Notebook, web application servers, and four graphical user interface toolkits. Matplotlib tries to make easy things easy and hard things possible. You can generate plots, histograms, power spectra, bar charts, error charts, scatter plots, etc., with just a few lines of code. For examples, see the sample plots and thumbnail gallery.

For simple plotting the pyplot module provides a 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.

Scikit – learn

Scikit-learn provides 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.

Python

Python is an interpreted high-level programming language for general-purpose programming. Created by Guido van Rossum and first released in 1991, Python has a design philosophy that emphasizes code readability, notably using significant whitespace.

Python features a dynamic type system and automatic memory management. It supports multiple programming paradigms, including object-oriented, imperative, functional and procedural, and has a large and comprehensive standard library.

- Python 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 also acknowledges that speed of development is important. Readable and terse code is part of this, and so is access to powerful constructs that avoid tedious repetition of code. Maintainability also ties into this may be an all but useless metric, but it does say something about how much code you have to scan, read and/or understand to troubleshoot problems or tweak behaviors. This speed of development, the ease with which a programmer of other languages can pick up basic Python skills and the huge standard library is key to another area where Python excels. All its tools have been quick to implement, saved a lot of time, and several of them have later been patched and updated by people with no Python background - without breaking.

5.5 INSTALL PYTHON STEP-BY-STEP IN WINDOWS AND MAC

Python a versatile programming language doesn't come pre-installed on your computer devices. Python was first released in the year 1991 and until today it is a very popular high-level programming language. Its style philosophy emphasizes code readability with its notable use of great whitespace.

The object-oriented approach and language construct provided by Python enables programmers to write both clear and logical code for projects. This software does not come pre-packaged with Windows.

How to Install Python on Windows and Mac :

There have been several updates in the Python version over the years. The question is how to install Python? It might be confusing for the beginner who is willing to start learning Python but this tutorial will solve your query. The latest or the newest version of Python is version

3.7.4 or in other words, it is Python 3.

Note: The python version 3.7.4 cannot be used on Windows XP or earlier devices.

Before you start with the installation process of Python. First, you need to know about your **System Requirements**. Based on your system type i.e. operating system and based processor, you must download the python version. My system type is a **Windows 64-bit operating system**. So the steps below are to install python version 3.7.4 on Windows 7 device or to install Python 3. Download the Python Cheat Sheet here. The steps on how to install Python on Windows 10, 8 and 7 are **divided into 4 parts** to help understand better.

Download the Correct version into the system

Step 1: Go to the official site to download and install python using Google Chrome or any other web browser. OR Click on the following link: <https://www.python.org>



Now, check for the latest and the correct version for your operating system.








Step 2: Click on the Download Tab.



Step 3: You can either select the Download Python for windows 3.7.4 button in Yellow Color or you can scroll further down and click on download with respective to their version. Here, we are downloading the most recent python version for windows 3.7.4

Looking for a specific release?

Python releases by version number:

| Release version | Release date | | Click for more |
|-------------------------------|----------------|--|-------------------------------|
| Python 3.7.4 | July 8, 2019 |  Download | Release Notes |
| Python 3.6.9 | July 2, 2019 |  Download | Release Notes |
| Python 3.7.3 | March 25, 2019 |  Download | Release Notes |
| Python 3.4.10 | March 18, 2019 |  Download | Release Notes |
| Python 3.5.7 | March 18, 2019 |  Download | Release Notes |
| Python 2.7.16 | March 4, 2019 |  Download | Release Notes |
| Python 3.7.2 | Dec. 24, 2018 |  Download | Release Notes |

Step 4: Scroll down the page until you find the Files option.

Step 5: Here you see a different version of python along with the operating system.

| Files | | | | | |
|---|------------------|-----------------------------|-----------------------------------|-----------|-----|
| Version | Operating System | Description | MD5 Sum | File Size | GP6 |
| Clipped source tarball | Source release | | 68111671e5b2db4ae709ab01b709be | 13017663 | 505 |
| X2 compressed source tarball | Source release | | d33e4aa6097051c2eca45ee3604803 | 17131432 | 505 |
| macOS 64-bit/32-bit installer | Mac OS X | for Mac OS X 10.5 and later | 6428b4675235a71a4c2c8a8e08e6 | 54898416 | 505 |
| macOS 64-bit installer | Mac OS X | for OS X 10.9 and later | 5d805c38217a45773b5e4a936b241f | 20882845 | 505 |
| Windows help file | Windows | | d63996573a2c5652ac58cad6b477cd2 | 8131761 | 505 |
| Windows x86-64 embeddable zip file | Windows | for AMD64/EM64/x64 | 9b093c8fd8ec8b5ab8c318aa0729a2 | 7504391 | 505 |
| Windows x86-64 executable installer | Windows | for AMD64/EM64/x64 | a702b4b0a778b8bd3043a383e5a3400 | 26883968 | 505 |
| Windows x86-64 web-based installer | Windows | for AMD64/EM64/x64 | 28c31c9088bd72ae8e53a3bd351b4bd2 | 1382904 | 505 |
| Windows x86 embeddable zip file | Windows | | 9fab3b818841879fda94132574139d8 | 6741628 | 505 |
| Windows x86 executable installer | Windows | | 33c3802942a54446a3b9451476394789 | 25663848 | 505 |
| Windows x86 web-based installer | Windows | | 1b670cfa5d3117df82c30983ea371d87c | 1324608 | 505 |

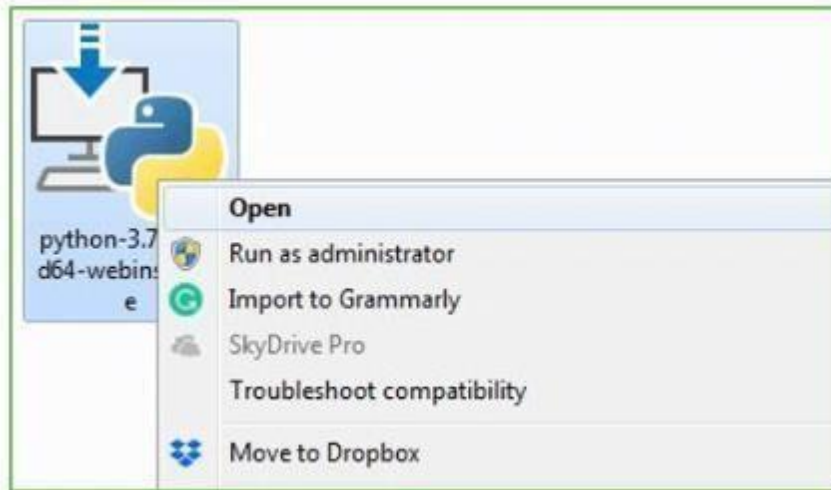
- To download Windows 32-bit python, you can select any one from the three options: Windows x86 embeddable zip file, Windows x86 executable installer or Windows x86 webbased installer.
- To download Windows 64-bit python, you can select any one from the three options: Windows x86-64 embeddable zip file, Windows x86-64 executable installer or Windows x8664 web-based installer.

Here we will install Windows x86-64 web-based installer. Here your first part regarding which version of python is to be downloaded is completed. Now we move ahead with the second part in installing python i.e. Installation

Note: To know the changes or updates that are made in the version you can click on the Release Note Option.

Installation of Python

Step 1: Go to Download and Open the downloaded python version to carry out the installation process.



Step 2: Before you click on Install Now, Make sure to put a tick on Add Python 3.7 to PATH.



Fig. 6

Step 3: Click on Install NOW After the installation is successful. Click on Close.

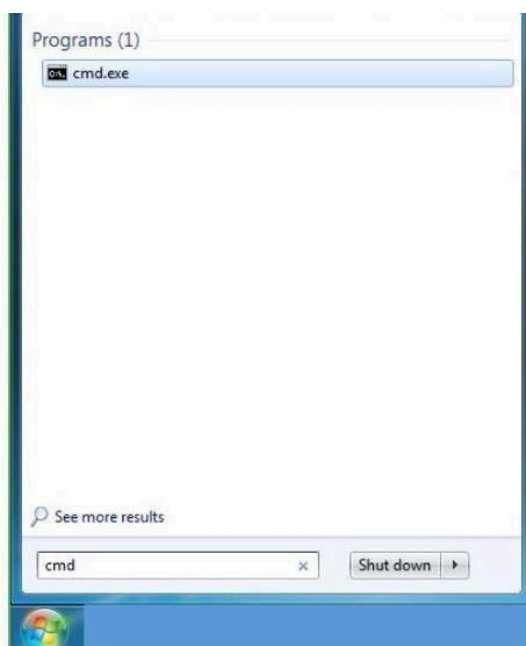


With these above three steps on python installation, you have successfully and correctly installed Python. Now is the time to verify the installation. **Note:** The installation process might take a couple of minutes.

Verify the Python Installation

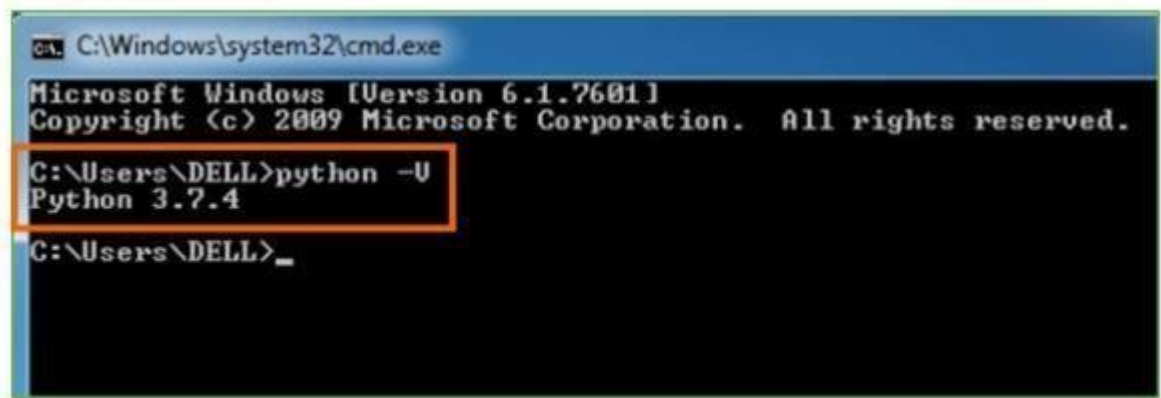
Step 1: Click on Start

Step 2: In the Windows Run Command, type “cmd”.



Step 3: Open the Command prompt option.

Step 4: Let us test whether the python is correctly installed. Type **python -V** and press Enter.



```
C:\Windows\system32\cmd.exe
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.

C:\Users\DELL>python -U
Python 3.7.4

C:\Users\DELL>_
```

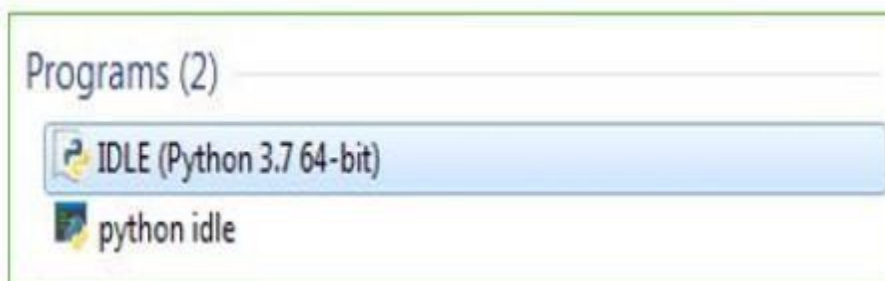
Step 5: You will get the answer as 3.7.4

Note: If you have any of the earlier versions of Python already installed. You must first uninstall the earlier version and then install the new one.

Check how the Python IDLE works

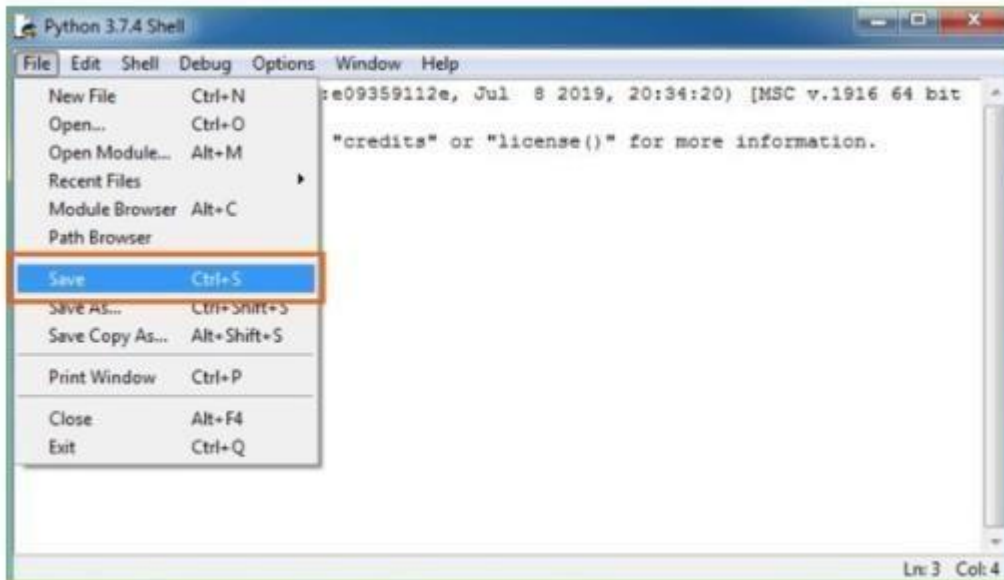
Step 1: Click on Start

Step 2: In the Windows Run command, type “python idle”.



Step 3: Click on IDLE (Python 3.7 64-bit) and launch the program

Step 4: To go ahead with working in IDLE you must first save the file. **Click on File > Click on Save**



Step 5: Name the file and save as type should be Python files. Click on SAVE. Here I have named the files as Hey World.

Step 6: Now for e.g. **enter print**

6. IMPLEMENTATIONS

6.1 SOFTWARE ENVIRONMENT

What is Python programming language?

Python is a **high-level, general-purpose, interpreted** programming language.

1) High-level

Python is a high-level programming language that makes it easy to learn. Python doesn't require you to understand the details of the computer in order to develop programs efficiently.

2) General-purpose

Python is a general-purpose language. It means that you can use Python in various domains including:

- Web applications
- Big data applications
- Testing
- Automation
- Data science, machine learning, and AI
- Desktop software
- Mobile apps

The targeted language like SQL which can be used for querying data from relational databases.

3) Interpreted

Python is an interpreted language. To develop a Python program, you write Python code into a file called source code. To execute the source code, you need to convert it to the machine language that the computer can understand. And the Python **interpreter** turns the source code, line by line, once at a time, into the machine code when the Python program executes.

Compiled languages like Java and C# use a **compiler** that compiles the whole source code before the program executes.

6.1.1 PYTHON

Python is a general-purpose interpreted, interactive, object-oriented, and high-level programming language. An interpreted language, Python has a design philosophy that emphasizes code readability (notably using whitespace indentation to delimit code blocks rather than curly brackets or keywords), and a syntax that allows programmers to express concepts in fewer lines of code than might be used in languages such as C++ or Java. It provides constructs that enable clear programming on both small and large scales. Python interpreters are available for many operating systems. CPython, the reference implementation of Python, is open source software and has a community-based development model, as do nearly all of its variant implementations. CPython is managed by the non-profit Python Software Foundation. Python features a dynamic type system and automatic memory management. Interactive Mode Programming.

Tkinter Programming

Tkinter is the standard GUI library for Python. Python when combined with Tkinter provides a fast and easy way to create GUI applications. Tkinter provides a powerful object-oriented interface to the Tk GUI toolkit. Creating a GUI application using Tkinter, we need to do is perform the following steps:

- Import the tkinter module.
- Create the GUI application main window.
- Add one or more widgets to the GUI application.
- Enter the main event loop to take action against each event triggered by the user.

Tkinter Widgets

tkinter provides various controls, such as buttons, labels and text boxes used in a GUI application. These controls are commonly called widgets.

6.1.2 SAMPLE CODE

Emotion

```
from tkinter import messagebox
from tkinter import *
from tkinter import simpledialog
import tkinter
from tkinter import filedialog
from imutils import paths
import numpy as np
from collections import defaultdict
from tkinter.filedialog import askopenfilename
from tkinter import simpledialog
from keras.preprocessing.image import img_to_array
from keras.models import load_model
import imutils
import cv2
import numpy as np
import sys
from tkinter import ttk
import os
from playsound import playsound

main = tkinter.Tk()
main.title("EMOTION BASED MUSIC RECOMMENDATION SYSTEM")
main.geometry("1200x1200")

global value
global filename
global faces
global frame
detection_model_path = 'haarcascade_frontalface_default.xml'
emotion_model_path = '_mini_XCEPTION.106-0.65.hdf5'
face_detection = cv2.CascadeClassifier(detection_model_path)
emotion_classifier = load_model(emotion_model_path, compile=False)
EMOTIONS = ["angry", "disgust", "scared", "happy", "sad", "surprised", "neutral"]

def upload():
    global filename
    global value
    filename = askopenfilename(initialdir = "images")
    pathlabel.config(text=filename)

def preprocess():
    global filename
    global frame
    global faces
    text.delete('1.0', END)
    orig_frame = cv2.imread(filename)
    orig_frame = cv2.resize(orig_frame, (48, 48))
```

```

frame = cv2.imread(filename,0)
faces =
face_detection.detectMultiScale(frame,scaleFactor=1.1,minNeighbors=5,minSize=(30,30),fl
ags=cv2.CASCADE_SCALE_IMAGE)
text.insert(END,"Total number of faces detected : "+str(len(faces)))

def detectEmotion():
    global faces
    if len(faces) > 0:
        faces = sorted(faces, reverse=True,key=lambda x: (x[2] - x[0]) * (x[3] - x[1]))[0]
        (fX, fY, fW, fH) = faces
        roi = frame[fY:fY + fH, fX:fX + fW]
        roi = cv2.resize(roi, (48, 48))
        roi = roi.astype("float") / 255.0
        roi = img_to_array(roi)
        roi = np.expand_dims(roi, axis=0)
        preds = emotion_classifier.predict(roi)[0]
        emotion_probability = np.max(preds)
        label = EMOTIONS[preds.argmax()]
        messagebox.showinfo("Emotion Prediction Screen", "Emotion Detected As : "+label)
        value.clear()
        path = 'songs'
        for r, d, f in os.walk(path):
            for file in f:
                if file.find(label) != -1:
                    value.append(file)
        else:
            messagebox.showinfo("Emotion Prediction Screen", "No face detected in uploaded
image")

def playSong():
    name = songslist.get()
    playsound('songs/'+name)

font = ('times', 20, 'bold')
title = Label(main, text='EMOTION-RESPONSIVE MUSIC SUGGESTION SYSTEM')
title.config(bg='brown', fg='white')
title.config(font=font)
title.config(height=3, width=80)
title.place(x=5,y=5)

font1 = ('times', 14, 'bold')
upload = Button(main, text="Upload Image With Face", command=upload)
upload.place(x=50,y=100)
upload.config(font=font1)

pathlabel = Label(main)
pathlabel.config(bg='brown', fg='white')
pathlabel.config(font=font1)
pathlabel.place(x=300,y=100)

```

```

preprocessbutton = Button(main, text="Preprocess & Detect Face in Image",
command=preprocess)
preprocessbutton.place(x=50,y=150)
preprocessbutton.config(font=font1)

emotion = Button(main, text="Detect Emotion", command=detectEmotion)
emotion.place(x=50,y=200)
emotion.config(font=font1)

emotionlabel = Label(main)
emotionlabel.config(bg='brown', fg='white')
emotionlabel.config(font=font1)
emotionlabel.place(x=610,y=200)
emotionlabel.config(text="Predicted Song")

value = ["Song List"]
songlist = ttk.Combobox(main, values=value, postcommand=lambda:
songlist.configure(values=value))
songlist.place(x=760,y=210)
songlist.current(0)
songlist.config(font=font1)

playsong = Button(main, text="Play Song", command=playSong)
playsong.place(x=50,y=250)
playsong.config(font=font1)

font1 = ('times', 12, 'bold')
text=Text(main,height=10,width=150)
scroll=Scrollbar(text)
text.configure(yscrollcommand=scroll.set)
text.place(x=10,y=300)
text.config(font=font1)

main.config(bg='brown')
main.mainloop()

```

Urls

```

from django.contrib import admin
from django.urls import path
from mainapp import views as mainapp_views
from userapp import views as userapp_views
from adminapp import views as adminapp_views
from django.conf import settings
from django.conf.urls.static import static

urlpatterns = [
    path('admin/', admin.site.urls),

```



```

#mainapp views
    path("",mainapp_views.main_index,name='main-index'),
    path('about',mainapp_views.about,name='about'),
    path('contact',mainapp_views.contact,name='contact'),

#userapp views
    path('user-login',userapp_views.user_login,name='user-login'),
    path('user-register',userapp_views.user_register,name='user-register'),
    path('user-index',userapp_views.user_index,name='user-index'),
    path('my-profile',userapp_views.my_profile,name='my-profile'),
    path('user-feedback',userapp_views.user_feedback,name='user-feedback'),
    path('galgobot',userapp_views.galgobot,name='galgobot'),

path('otp-verification/<int:id>',userapp_views.user_otpverification,name='user-otpverification'
),

#adminapp views
    path('admin-login',adminapp_views.admin_login,name='admin-login'),
    path('admin-index',adminapp_views.admin_index,name='admin-index'),
    path('admin-interactions',adminapp_views.admin_interactions,name='admin-interactions'),

path('admin-sentiment-analy',adminapp_views.admin_sentiment_analy,name='admin-sentime
nt-analy'),

]
urlpatterns+=static(settings.MEDIA_URL,document_root=settings.MEDIA_ROOT)

```

Fee structure

```

{
    "id": "1fa7b64c-7f0c-4a6d-b352-85e1e35855af",
    "name": "BTech-Fees-Structure",
    "auto": true,
    "contexts": [],
    "responses": [
        {
            "resetContexts": false,
            "action": "",
            "affectedContexts": [],
            "parameters": [],
            "messages": [
                {
                    "type": "0",
                    "title": "",
                    "textToSpeech": "",
                    "lang": "en",
                    "speech": [
                        "BTech-Fees-Structure:\n\nBranches : All Branches\nDuration : 4
Years\nFee-Payment : 5,50,0000 for 4 years"
                    ],
                    "condition": ""
                }
            ]
        }
    ]
}

```

```

    },
    {
      "type": "4",
      "title": "",
      "payload": {
        "richText": [
          [
            {
              "options": [
                {
                  "value": "Select Topic",
                  "name": "Select Topic",
                  "type": "button",
                  "image": {
                    "src": {
                      "rawUrl": "https://cdn-icons-png.flaticon.com/512/3240/3240831.png"
                    }
                  },
                  "text": "Select Topic"
                }
              ],
              "type": "chips"
            }
          ]
        ],
        "textToSpeech": "",
        "lang": "en",
        "condition": ""
      }
    },
    "speech": []
  }
}

```

Start chat

```

{
  "id": "a90944fa-549b-4c06-86d5-72d32a12b98a",
  "name": "Start-Chat",
  "auto": true,
  "contexts": [],
  "responses": [
    {
      "resetContexts": false,
      "action": "",
      "affectedContexts": [],
      "parameters": [],
      "messages": [
        {
          "type": "0",
          "title": "",
          "textToSpeech": "",

```

```

    "lang": "en",
    "condition": ""
  },
  {
    "type": "4",
    "title": "",
    "payload": {
      "richText": [
        [
          {
            "type": "chips",
            "options": [
              {
                "type": "button",
                "name": "Admission-Details",
                "value": "Admission-Details",
                "image": {
                  "src": {
                    "rawUrl": "https://cdn-icons-png.flaticon.com/512/3240/3240831.png"
                  }
                },
                "text": "Admission-Details"
              },
              {
                "type": "button",
                "image": {
                  "src": {
                    "rawUrl": "https://cdn-icons-png.flaticon.com/512/3240/3240831.png"
                  }
                },
                "value": "Fee-Structure",
                "name": "Fee-Structure",
                "text": "Fee-Structure"
              },
              {
                "type": "button",
                "text": "Courses",
                "name": "Courses",
                "image": {
                  "src": {
                    "rawUrl": "https://cdn-icons-png.flaticon.com/512/3240/3240831.png"
                  }
                },
                "value": "Courses"
              },
              {
                "value": "Departments",
                "image": {
                  "src": {
                    "rawUrl": "https://cdn-icons-png.flaticon.com/512/3240/3240831.png"
                  }
                }
              },
            ]
          }
        ]
      }
    }
  },

```

```

        "type": "button",
        "text": "Departments",
        "name": "Departments"
    },
    {
        "value": "Faculties",
        "image": {
            "src": {
                "rawUrl": "https://cdn-icons-png.flaticon.com/512/3240/3240831.png"
            }
        }
    },

    {
        "value": "Placements",
        "name": "Placements",
        "text": "Placements",
        "image": {
            "src": {
                "rawUrl": "https://cdn-icons-png.flaticon.com/512/3240/3240831.png"
            }
        }
    },
    {
        "type": "button"
    },
    {
        "value": "Events",
        "name": "Events",
        "type": "button",
        "image": {
            "src": {
                "rawUrl": "https://cdn-icons-png.flaticon.com/512/3240/3240831.png"
            }
        }
    },
    {
        "rawUrl": "https://cdn-icons-png.flaticon.com/512/3240/3240831.png"
    }
],
"priority": 500000,
"webhookUsed": false,
"webhookForSlotFilling": false,
"fallbackIntent": false,
"events": [],
"conditionalResponses": [],
"condition": "",
"conditionalFollowupEvents": []
}

```

Workshops

```

{
    "id": "2fc62386-d21d-4466-a182-962f99614f86",
    "name": "Workshops",
    "auto": true,
    "contexts": [],
    "responses": [
        {

```

```

"resetContexts": false,
"action": "",
"affectedContexts": [],
"parameters": [],
"messages": [
  {
    "type": "0",
    "title": "",

    {
      "type": "4",
      "title": "",
      "payload": {
        "richText": [
          [

            {
              "type": "4",
              "title": "",
              "payload": {
                "richText": [
                  [
                    {
                      "options": [
                        {
                          "name": "Select Topic",
                          "value": "Select Topic",
                          "text": "Select Topic",
                          "type": "button",
                          "image": {
                            "src": {
                              "rawUrl": "https://cdn-icons-png.flaticon.com/512/3240/3240831.png"
                            }
                          }
                        }
                      ]
                    }
                  ]
                }
              }
            }
          ]
        ]
      }
    }
  ],
  "priority": 500000,
  "webhookUsed": false,
  "webhookForSlotFilling": false,
  "fallbackIntent": false,
  "events": [],
  "conditionalResponses": [],
  "condition": "",
  "conditionalFollowupEvents": []
}

```

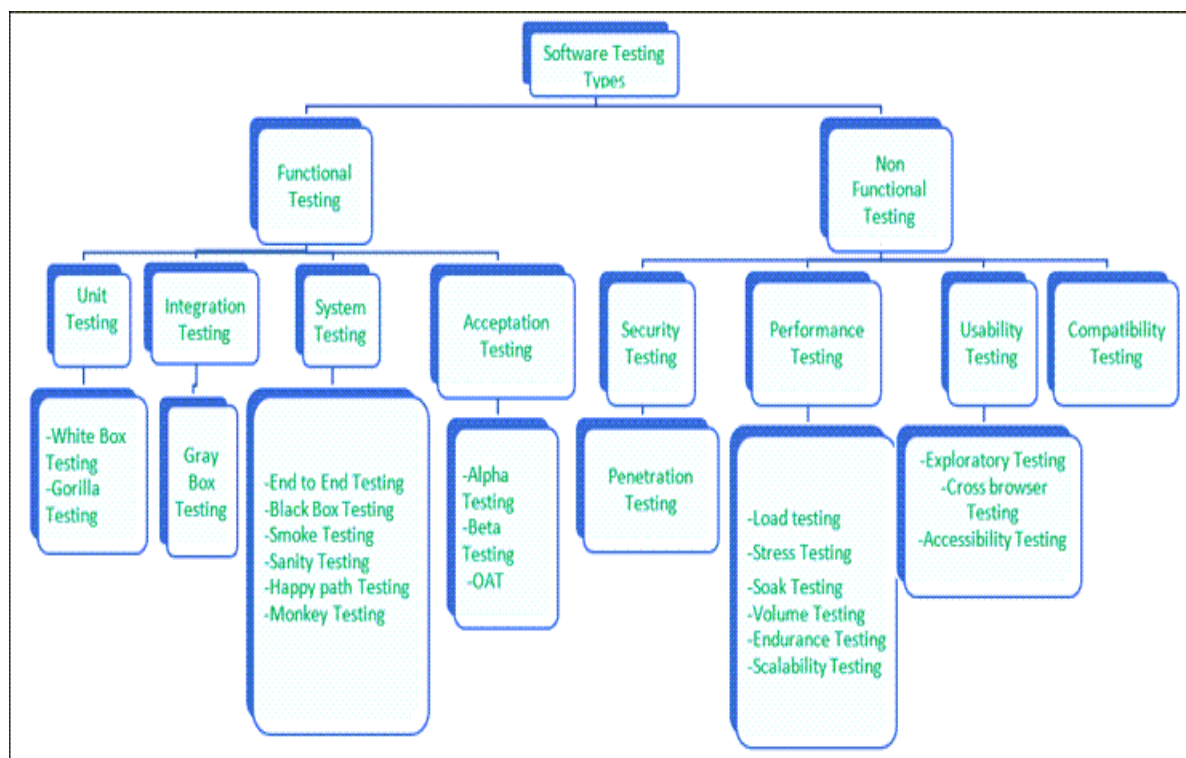
7.SYSTEM TESTING

7.1 INTRODUCTION TO TESTING

Types of Software Testing: Different Testing Types with Details

We, as testers, are aware of the various types of Software Testing like Functional Testing, Non-Functional Testing, Automation Testing, Agile Testing, and their sub-types, etc. Each type of testing has its own features, advantages, and disadvantages as well. However, in this tutorial, we have covered mostly each and every type of software testing which we usually use in our day-to-day testing life.

Different Types of Software Testing



7.2 TESTING STRATEGIES

Functional testing

There are four main types of functional testing.

1) Unit Testing

Unit testing is a type of software testing which is done on an individual unit or component to test its corrections. Typically, Unit testing is done by the developer at the application development phase. Each unit in unit testing can be viewed as a method, function, procedure, or object. Developers often use test automation tools such as NUnit, Xunit, JUnit for the test execution. Unit testing is important because we can find more defects at the unit test level.

For example, there is a simple calculator application. The developer can write the unit test to check if the user can enter two numbers and get the correct sum for addition functionality.

a)White Box Testing:

White box testing is a test technique in which the internal structure or code of an application is visible and accessible to the tester. In this technique, it is easy to find loopholes in the design of an application or fault in business logic. Statement coverage and decision coverage/branch coverage are examples of white box test techniques.

b)Gorilla Testing:

Gorilla testing is a test technique in which the tester and/or developer test the module of the application thoroughly in all aspects. Gorilla testing is done to check how robust your application is.

For example, the tester is testing the pet insurance company's website, which provides the service of buying an insurance policy, tag for the pet, Lifetime membership. The tester can focus on any one module, let's say, the insurance policy module, and test it thoroughly with positive and negative test scenarios.

2) Integration Testing

Integration testing is a type of software testing where two or more modules of an application are logically grouped together and tested as a whole. The focus of this type of testing is to find the defect on interface, communication, and data flow among modules. Top-down or Bottom-up approach is used while integrating modules into the whole system.

This type of testing is done on integrating modules of a system or between systems.

For example, a user is buying a flight ticket from any airline website. Users can see flight details and payment information while buying a ticket, but flight details and payment processing are two different systems. Integration testing should be done while integrating of airline website and payment processing system.

a) Gray box testing

As the name suggests, gray box testing is a combination of white-box testing and black-box testing. Testers have partial knowledge of the internal structure or code of an application.

System Testing

System testing is types of testing where tester evaluates the whole system against the specified requirements.

a) End to End Testing

It involves testing a complete application environment in a situation that mimics real-world use, such as interacting with a database, using network communications, or interacting with other hardware, applications, or systems if appropriate.

For example, a tester is testing a pet insurance website. End to End testing involves testing of buying an insurance policy, LPM, tag, adding another pet, updating credit card information on users' accounts, updating user address information, receiving order confirmation emails and policy documents.

b) Black Box Testing

Black box testing is a software testing technique in which testing is performed without knowing the internal structure, design, or code of a system under test. Testers should focus only on the input and output of test objects. Detailed information about the advantages, disadvantages, and types of Black Box testing can be found [here](#).

c) Smoke Testing

Smoke testing is performed to verify that basic and critical functionality of the system under test is working fine at a very high level. Whenever a new build is provided by the development team, then the Software Testing team validates the build and ensures that no major issue exists. The testing team will ensure that the build is stable, and a detailed level of testing will be carried out further.

For example, tester is testing pet insurance website. Buying an insurance policy, adding another pet, providing quotes are all basic and critical functionality of the application. Smoke testing for this website verifies that all these functionalities are working fine before doing any in-depth testing.

d) Sanity Testing

Sanity testing is performed on a system to verify that newly added functionality or bug fixes are working fine. Sanity testing is done on stable build. It is a subset of the regression test.

For example, a tester is testing a pet insurance website. There is a change in the discount for buying a policy for second pet. Then sanity testing is only performed on buying insurance policy module.

e) Happy path Testing

The objective of Happy Path Testing is to test an application successfully on a positive flow. It does not look for negative or error conditions. The focus is only on valid and positive inputs through which the application generates the expected output.

f) Monkey Testing

Monkey Testing is carried out by a tester, assuming that if the monkey uses the application, then how random input and values will be entered by the Monkey without any knowledge or understanding of the application. The objective of Monkey Testing is to check if an application or system gets crashed by providing random input values/data. Monkey Testing is performed randomly, no test cases are scripted, and it is not necessary to be aware of the full functionality of the system.

4) Acceptance Testing

Acceptance testing is a type of testing where client/business/customer test the software with real time business scenarios. The client accepts the software only when all the features and functionalities work as expected. This is the last phase of testing, after which the software goes into production. This is also called User Acceptance Testing (UAT).

a) Alpha Testing

Alpha testing is a type of acceptance testing performed by the team in an organization to find as many defects as possible before releasing software to customers.

For example, the pet insurance website is under UAT. UAT team will run real-time scenarios like buying an insurance policy, buying annual membership, changing the address, ownership transfer of the pet in a same way the user uses the real website. The team can use test credit card information to process payment-related scenarios.

b) Beta Testing

Beta Testing is a type of software testing which is carried out by the clients/customers. It is performed in the Real Environment before releasing the product to the market for the actual end-users. Beta Testing is carried out to ensure that there are no major failures in the software or product, and it satisfies the business requirements from an end-user perspective.

Beta Testing is successful when the customer accepts the software. Usually, this testing is typically done by the end-users. This is the final testing done before releasing the application for commercial purposes. Usually, the Beta version of the software or product released is limited to a certain number of users in a specific area. So, the end-user uses the software and shares the feedback with the company. The company then takes necessary action before releasing the software worldwide.

c) Operational acceptance testing (OAT)

Operational acceptance testing of the system is performed by operations or system administration staff in the production environment. The purpose of operational acceptance testing is to make sure that the system administrators can keep the system working properly for the users in a real-time environment.

The focus of the OAT is on the following points:

- Testing of backup and restore.
- Installing, uninstalling, upgrading software.
- The recovery process in case of natural disaster.
- User management.
- Maintenance of the software.

Non-Functional Testing:

There are four main types of functional testing.

1) Security Testing

It is a type of testing performed by a special team. Any hacking method can penetrate the system. Security Testing is done to check how the software, application, or website is secure from internal and/or external threats. This testing includes how much software is secure from malicious programs, viruses and how secure & strong the authorization and authentication processes are. It also checks how software behaves for any hacker's attack & malicious programs and how software is maintained for data security after such a hacker attack.

a) Penetration Testing

Penetration Testing or Pen testing is the type of security testing performed as an authorized cyberattack on the system to find out the weak points of the system in terms of security. Pen testing is performed by outside contractors, generally known as ethical hackers. That is why it is also known as ethical hacking.

Contractors perform different operations like SQL injection, URL manipulation, Privilege Elevation, session expiry, and provide reports to the organization.

Notes: Do not perform the Pen testing on your laptop/computer. Always take written permission to do pen tests.

2) Performance Testing

Performance testing is testing of an application's stability and response time by applying load. The word stability means the ability of the application to withstand in the presence of load. Response time is how quickly an application is available to users. Performance testing is done with the help of tools. Loader.IO, JMeter, LoadRunner, etc. are good tools available in the market.

a) Load testing

Load testing is testing of an application's stability and response time by applying load, which is equal to or less than the designed number of users for an application.

For example, your application handles 100 users at a time with a response time of 3 seconds, then load testing can be done by applying a load of the maximum of 100 or less than 100 users. The goal is to verify that the application is responding within 3 seconds for all the users.

b) Stress Testing

Stress testing is testing an application's stability and response time by applying load, which is more than the designed number of users for an application.

For example, your application handles 1000 users at a time with a response time of 4 seconds, then stress testing can be done by applying a load of more than 1000 users. Test the application with 1100,1200,1300 users and notice the response time. The goal is to verify the stability of an application under stress.

c) Scalability Testing

Scalability testing is testing an application's stability and response time by applying load, which is more than the designed number of users for an application.

For example, your application handles 1000 users at a time with a response time of 2 seconds, then scalability testing can be done by applying a load of more than 1000 users and gradually increasing the number of users to find out where exactly my application is crashing.

Let's say my application is giving response time as follows:

- 1000 users -2 sec
- 1400 users -2 sec
- 4000 users -3 sec
- 5000 users -45 sec
- 5150 users- crash – This is the point that needs to identify in scalability testing

d) Volume testing (flood testing)

Volume testing is testing an application's stability and response time by transferring a large volume of data to the database. Basically, it tests the capacity of the database to handle the data.

e) Endurance Testing (Soak Testing)

Endurance testing is testing an application's stability and response time by applying load continuously for a longer period to verify that the application is working fine.

For example, car companies soak testing to verify that users can drive cars continuously for hours without any problem.

3) Usability Testing

Usability testing is testing an application from the user's perspective to check the look and feel and user-friendliness.

For example, there is a mobile app for stock trading, and a tester is performing usability testing. Testers can check the scenario like if the mobile app is easy to operate with one hand or not, scroll bar should be vertical, background colour of the app should be black and price of and stock is displayed in red or green colour. The main idea of usability testing of this kind of app is that as soon as the user opens the app, the user should get a glance at the market.

a) Exploratory testing

Exploratory Testing is informal testing performed by the testing team. The objective of this testing is to explore the application and look for defects that exist in the application. Testers use the knowledge of the business domain to test the application. Test charters are used to guide the exploratory testing.

b) Cross browser testing

Cross browser testing is testing an application on different browsers, operating systems, mobile devices to see look and feel and performance. Why do we need cross-browser testing? The answer is different users use different operating systems, different browsers, and different mobile devices. The goal of the company is to get a good user experience regardless of those devices. Browser stack provides all the versions of all the browsers and all mobile devices to test the application. For learning purposes, it is good to take the free trial given by browser stack for a few days.

c) Accessibility Testing

The aim of Accessibility Testing is to determine whether the software or application is accessible for disabled people or not. Here, disability means deafness, colour blindness, mentally disabled, blind, old age, and other disabled groups.

Various checks are performed, such as font size for visually disabled, colour and contrast for colour blindness, etc.

4) Compatibility testing

This is a testing type in which it validates how software behaves and runs in a different environment, web servers, hardware, and network environment.

8.SCREENSHOTS

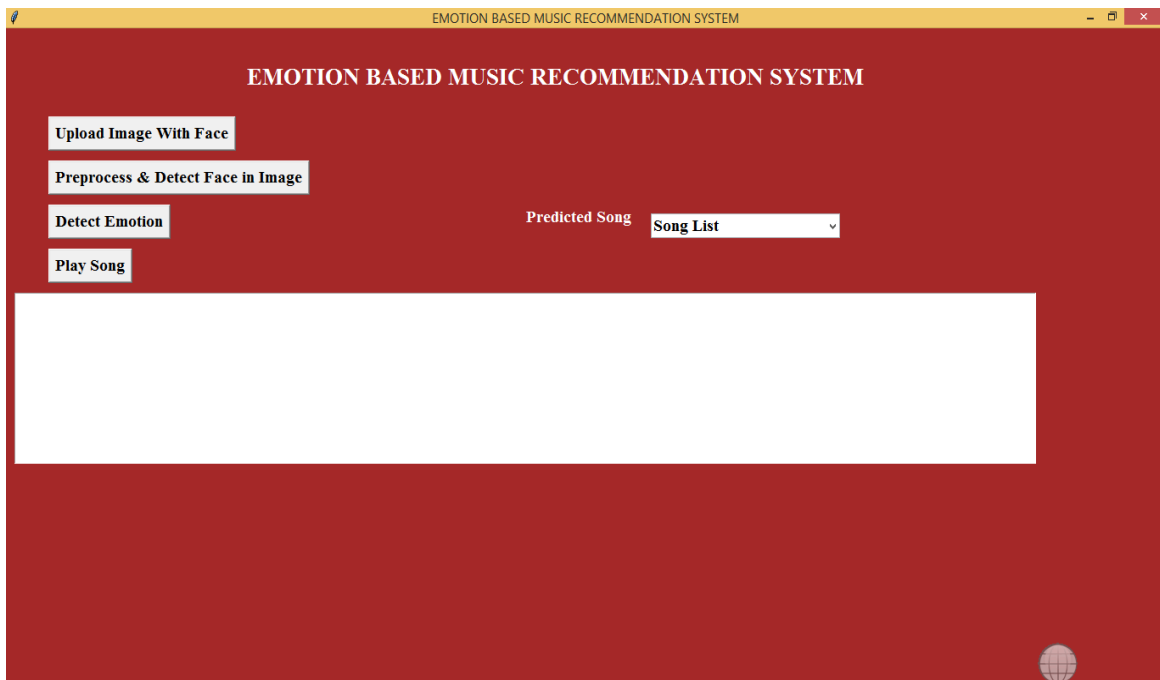


Fig. 1

In above screen click on 'Upload Image With Face' button to upload image

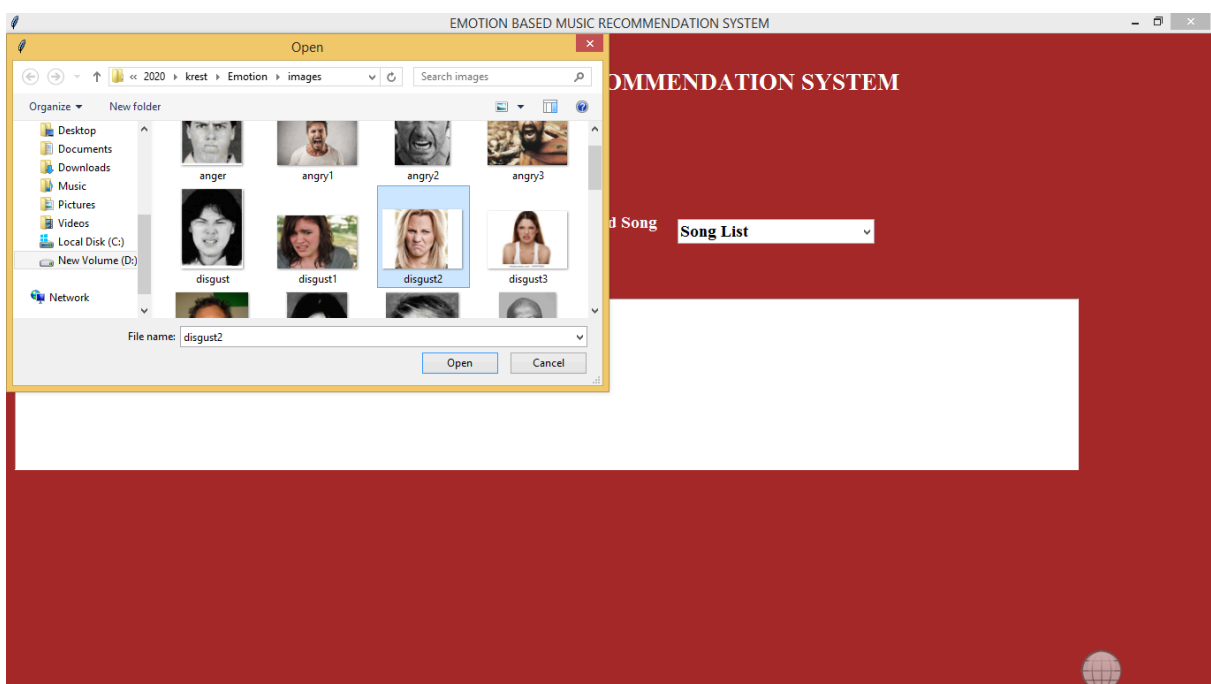


Fig. 2

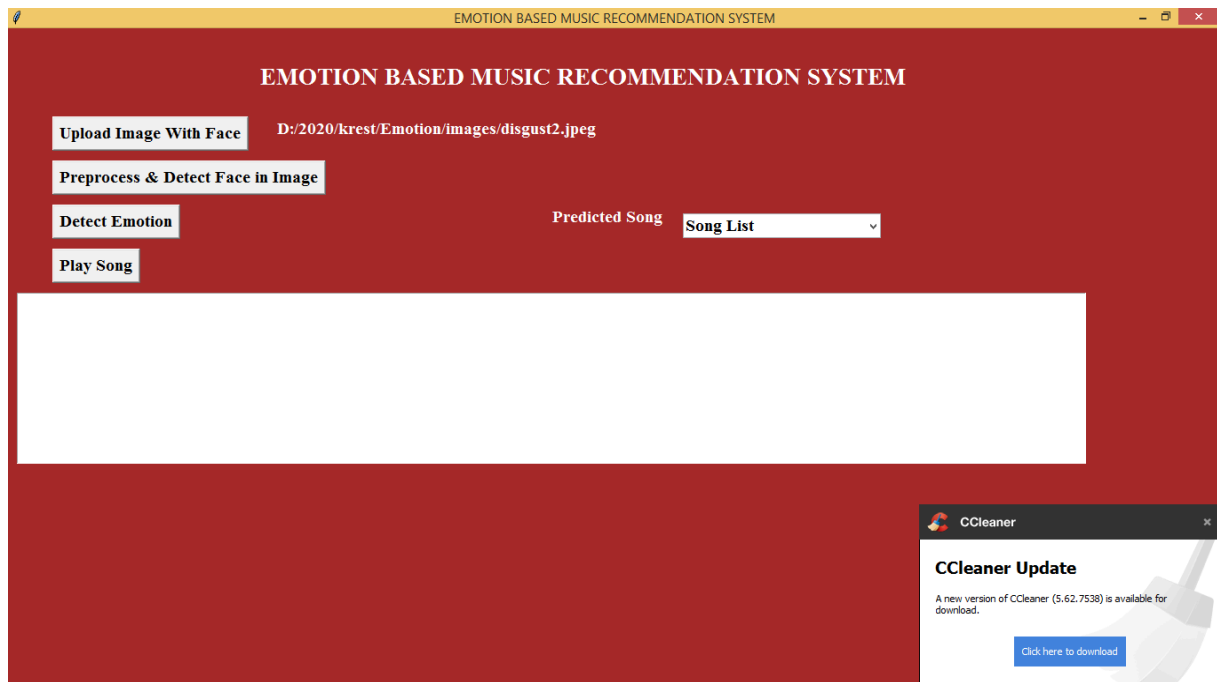


Fig. 3

In above screen i am selecting one 'disgust' image. Now click on 'Pre-process & Detect Face In Image' button to perform pre-processing and to extract face from images

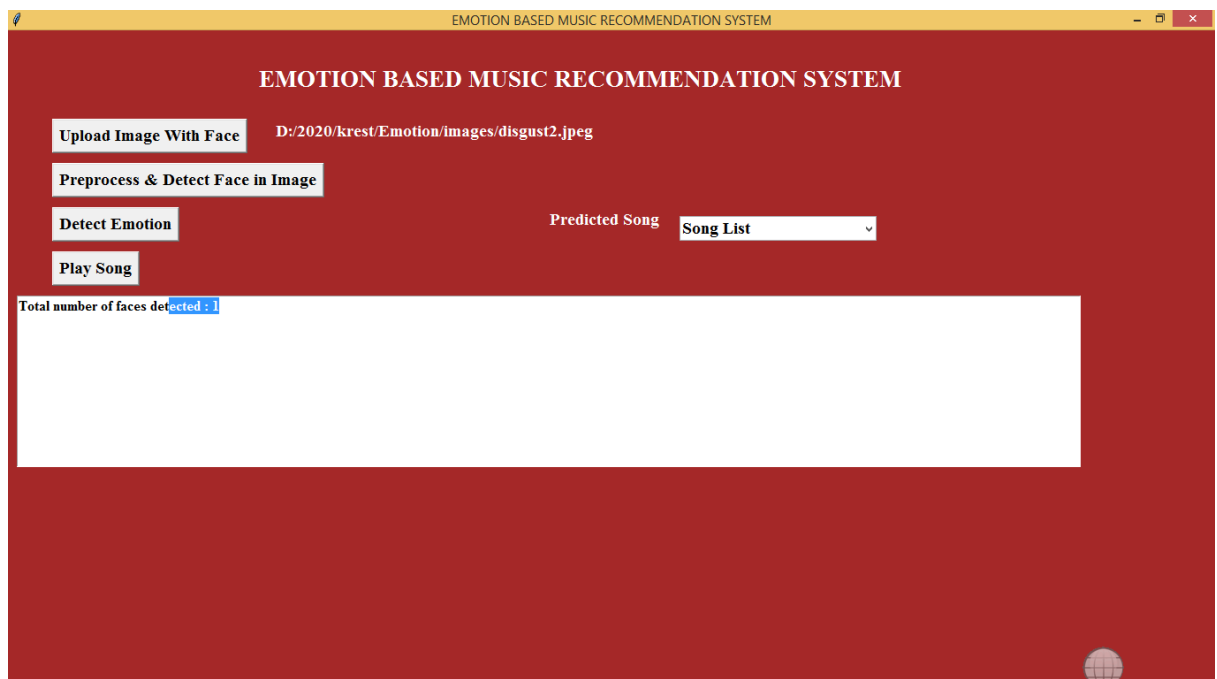


Fig. 4

In above screen we can see in uploaded image one face is detected. Now click on Detect Emotion button to detect emotion

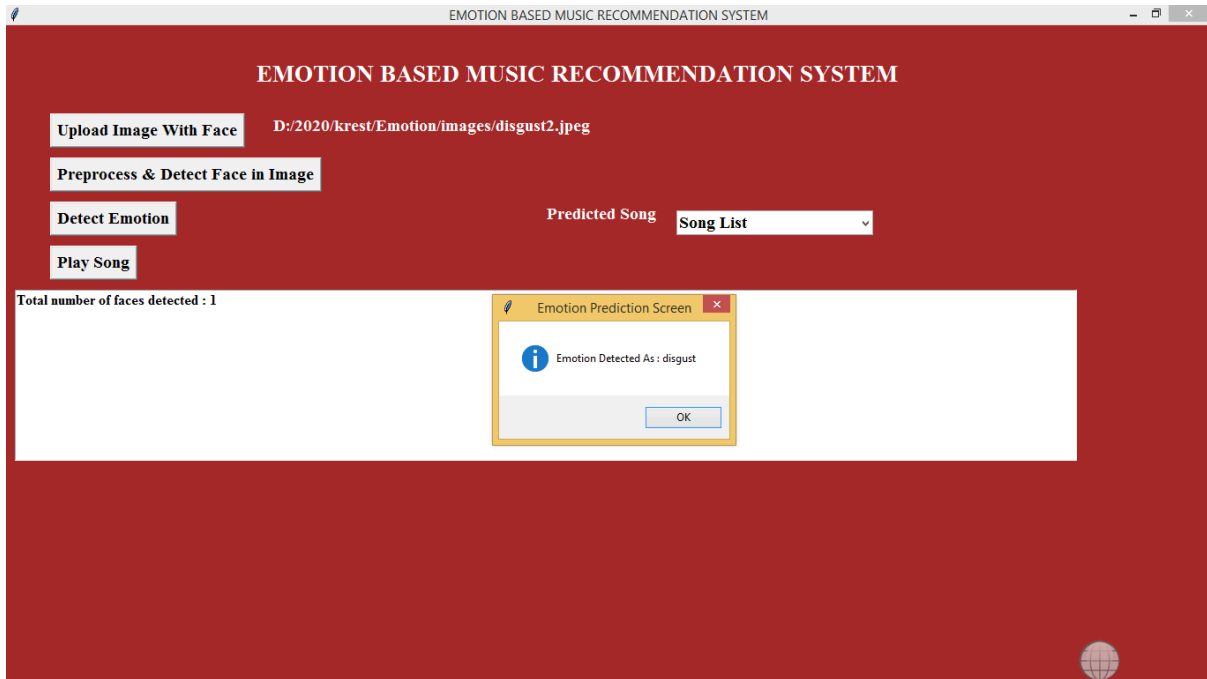


Fig. 5

In above screen we can see emotion disgust is detected and now click on drop down arrow link to get all disgust songs list

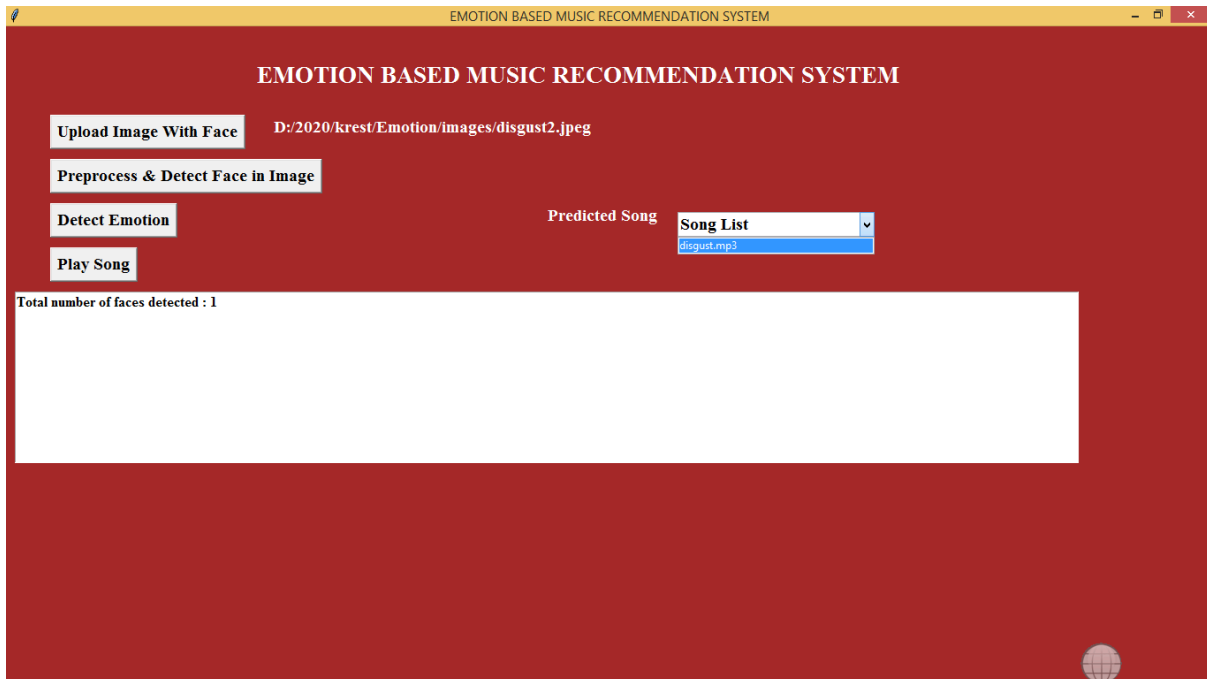


Fig. 6

In drop down box we can see 'disgust.mp3' songs is showing, select that song and click on 'Play Song' button to play song

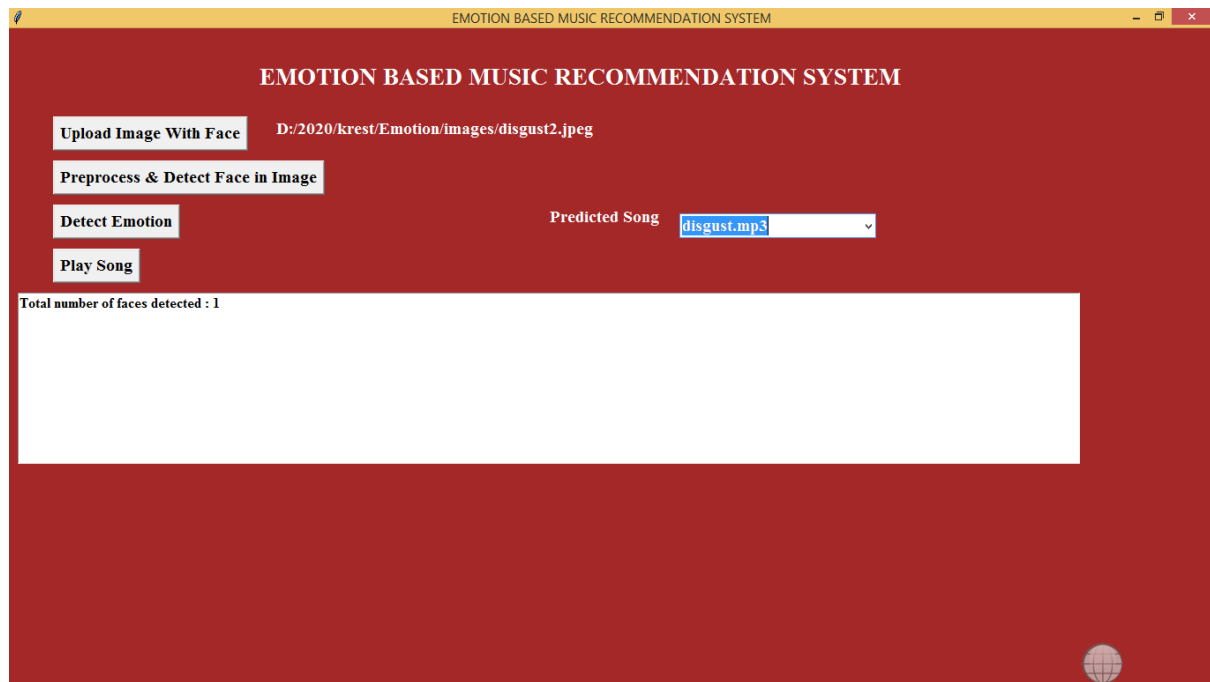


Fig. 7

If your system has audio driver then u can hear song.

Note: no technologies can detect 100% emotion from images but this project can detect upto 92%

9. CONCLUSIONS

In this study, a framework for enhancing music recommendation engines performance via physiological signals has been introduced. Emotion recognition from multi-channel physiological signals was performed, data fusion techniques were applied to combine data from GSR and PPG sensors and FLF has been implemented. Considering emotion state of the listener improves the performance of recommendations. Recognizing arousal and valence values directly from only GSR and PPG signals is a challenging task. We have showed that there is relationship between GSR and PPG signals and emotional arousal and valence dimensions. For GSR only signal, we have obtained 71.53% and 71.04% accuracy rate for arousal and valence prediction respectively. For photoplethysmography only signal, we have obtained 70.93% and 70.76% accuracy rate for arousal and valence prediction respectively. Fusing GSR and PPG signals we have obtained the results, 72.06% and 71.05% accuracy rate for arousal and valence prediction respectively. Although there is only slight improvement using fusion in emotion recognition accuracy, the proposed framework is promising for music recommendation engines in terms of adding multi modal emotion phenomenon into music recommendation logic. Performance can be improved with the advancement of wearable sensor technologies and using different type of sensors. Using more than one sensor may also help for failure management. As future work, we will consider different combination of sensors that handle the failures of wearable sensors and additional sensors usage to increase performance. The results of this study can be used to increase user experience of multimedia tools and music recommendation engines. Since there is high correlation between physiological GSR and PPG data and affective state and cognitive state of a person multimedia recommendation engines can benefit from physiological computing systems.

10. REFERENCES

- [1] S. Jhahharia, S. Pal, and S. Verma, “Wearable computing and its application,” *Int. J. Comp. Sci. and Inf. Tech.*, vol. 5, no. 4, pp. 5700– 5704, 2014.
- [2] K. Popat and P. Sharma, “Wearable computer applications: A feature perspective,” *Int. J. Eng. and Innov. Tech.*, vol. 3, no. 1, 2013.
- [3] P. Melville and V. Sindhwani, “Recommender systems,” in *Encyc. of mach. learn.* Springer, 2011, pp. 829–838.
- [4] N. Sebe, I. Cohen, T. S. Huang et al., “Multimodal emotion recognition,” *Handbook of Pattern Recognition and Computer Vision*, vol. 4, pp. 387– 419, 2005.
- [5] R. W. Picard, E. Vyzas, and J. Healey, “Toward machine emotional intelligence: Analysis of affective physiological state,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 23, no. 10, pp. 1175–1191, 2001.
- [6] D. Ayata, Y. Yaslan, and M. Kamasak, “Emotion recognition via galvanic skin response: Comparison of machine learning algorithms and feature extraction methods,” *IU J. of Elect. & Elect. Eng.*, vol. 17, no. 1, pp. 3129–3136, 2017.
- [7] P. Ekman, R. W. Levenson, and W. V. Friesen, “Autonomic nervous system activity distinguishes among emotions.” *Am. Assoc. for Adv. of Sci.*, 1983.
- [8] I.-h. Shin, J. Cha, G. W. Cheon, C. Lee, S. Y. Lee, H.-J. Yoon, and H. C. Kim, “Automatic stress-relieving music recommendation system based on photoplethysmography-derived heart rate variability analysis,” in *IEEE Int. Conf. on Eng. in Med. and Bio. Soc. IEEE*, 2014, pp. 6402–6405.
- [9] S. Nirjon, R. F. Dickerson, Q. Li, P. Asare, J. A. Stankovic, D. Hong, B. Zhang, X. Jiang, G. Shen, and F. Zhao, “Musicalheart: A hearty way of listening to music,” in *Proc. of ACM Conf. on Emb. Netw. Sens. Sys.* ACM, 2012, pp. 43–56.
- [10] H. Liu, J. Hu, and M. Rauterberg, “Music playlist recommendation based on user heartbeat and music preference,” in *Int. Conf. on Comp. Tech. and Dev.*, vol. 1. IEEE, 2009, pp. 545–549.
- [11] F. Isinkaye, Y. Folajimi, and B. Ojokoh, “Recommendation systems: Principles, methods

and evaluation,” *Egypt. Inf. J.*, vol. 16, no. 3, pp. 261–273, 2015.

[12] A. Nakasone, H. Prendinger, and M. Ishizuka, “Emotion recognition from electromyography and skin conductance,” in *Proc. of Int. Work. on Biosignal Interp.*, 2005, pp. 219–222.

[13] K. Yoon, J. Lee, and M. U. Kim, “Music recommendation system using emotion triggering low-level features,” *IEEE Trans. Consum. Electron*, vol. 58, no. 2, pp. 612–618, May 2012.

[14] R. L. Rosa, D. Z. Rodriguez, and G. Bressan, “Music recommendation system based on user’s sentiments extracted from social networks,” *IEEE Trans. Consum. Electron*, vol. 61, no. 3, pp. 359–367, Aug 2015.