

**A**  
**MAJOR PROJECT REPORT**  
**On**  
**Smart Civilization for Detecting Littering on Roads**  
**BACHELOR OF TECHNOLOGY**  
**In**  
**COMPUTER SCIENCE AND ENGINEERING**

**Submitted by**  
**(MIP-B10)**

**Student Name:**

**197Y5A0515 K.THARUN**

**197Y5A0510 T.RAJU SWAROOP**

**Under the Guidance**

**of**

**Mr.Ch.V.V. NARASIMHA RAJU**  
**Assistant Professor**



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**  
**MARRI LAXMAN REDDY**  
**INSTITUTE OF TECHNOLOGY AND MANAGEMENT**  
**(AUTONOMOUS)**

(Affiliated to JNTU-H, Approved by AICTE New Delhi and Accredited by NBA & NAAC With 'A' Grade)

**July 2022**



# **MARRI LAXMAN REDDY** **INSTITUTE OF TECHNOLOGY AND MANAGEMENT**

**(AN AUTONOMOUS INSTITUTION)**

(Approved by AICTE, New Delhi & Affiliated to JNTUH, Hyderabad)

Accredited by NBA and NAAC with 'A' Grade & Recognized Under Section 2(f) & 12(B) of the UGC act, 1956

## **CERTIFICATE**

---

This is to certify that the project report titled “**Smart Civilization for Detecting Littering on Roads**” is being submitted by **T. RAJU SWAROOP (197Y5A0510)** and **K. THARUN (197Y5A0515)** in IV B.Tech II Semester **Computer Science & Engineering** is a record bonafide work carried out by him. The results embodied in this report have not been submitted to any other University for the award of any degree.

**Internal Guide**

(Mr.Ch.V.V. NARASIMHA RAJU)

**HOD**

**Principal**

**External Examiner**



# **MARRI LAXMAN REDDY** **INSTITUTE OF TECHNOLOGY AND MANAGEMENT**

(AN AUTONOMOUS INSTITUTION)

(Approved by AICTE, New Delhi & Affiliated to JNTUH, Hyderabad)

Accredited by NBA and NAAC with 'A' Grade & Recognized Under Section 2(f) & 12(B) of the UGC act, 1956

## **DECLARATION**

---

We hereby declare that the Mini Project Report entitled, “**Smart Civilization For Detecting Littering on Roads**” submitted for the B.Tech degree is entirely our work and all ideas and references have been duly acknowledged. It does not contain any work for the award of any other degree.

**Date:**

**SWAROOP**  
**(197Y5A0510)**

**THARUN**  
**(197Y5A0515)**



# **MARRI LAXMAN REDDY** **INSTITUTE OF TECHNOLOGY AND MANAGEMENT**

(AN AUTONOMOUS INSTITUTION)

(Approved by AICTE, New Delhi & Affiliated to JNTUH, Hyderabad)

Accredited by NBA and NAAC with 'A' Grade & Recognized Under Section 2(f) & 12(B) of the UGC act, 1956

## **ACKNOWLEDGEMENT**

---

We are happy to express my deep sense of gratitude to the principal of the college **Dr. K. Venkateswara Reddy**, Professor, Department of Computer Science and Engineering, Marri Laxman Reddy Institute of Technology & Management, for having provided us with adequate facilities to pursue our project.

We would like to thank **Mr. Abdul Basith Khateeb**, Assoc. Professor and Head, Department of Computer Science and Engineering, Marri Laxman Reddy Institute of Technology & Management, for having provided the freedom to use all the facilities available in the department, especially the laboratories and the library.

We are very grateful to my project guide **Mr. Ch. V. V. Narasimha Raju**, Assoc. Prof., Department of Computer Science and Engineering, Marri Laxman Reddy Institute of Technology & Management, for his extensive patience and guidance throughout our project work.

We sincerely thank our seniors and all the teaching and non-teaching staff of the Department of Computer Science for their timely suggestions, healthy criticism and motivation during the course of this work.

We would also like to thank my classmates for always being there whenever we needed help or moral support. With great respect and obedience, we thank our parents and siblings who were the backbone behind our deeds.

Finally, we express our immense gratitude with pleasure to the other individuals who have either directly or indirectly contributed to our need at right time for the development and success of this work.



# **MARRI LAXMAN REDDY** **INSTITUTE OF TECHNOLOGY AND MANAGEMENT**

(AN AUTONOMOUS INSTITUTION)

(Approved by AICTE, New Delhi & Affiliated to JNTUH, Hyderabad)

Accredited by NBA and NAAC with 'A' Grade & Recognized Under Section 2(f) & 12(B) of the UGC act, 1956

## **CONTENTS**

<b>S NO.</b>	<b>TITLE</b>	<b>PAGE NO.</b>
	ABSTRACT	<b>Vi</b>
	LIST OF FIGURES	<b>Vii</b>
	LIST OF TABLES	<b>Viii</b>
	SYMBOLS & ABBREVIATIONS	<b>IX</b>
<b>1</b>	<b>INTRODUCTION</b>	<b>1</b>
	1.1 Purpose	1
	1.2 Scope	2
	1.3 Features	2
	1.4 Problem Definition	2
	1.5 Organization Of Document	3
<b>2</b>	<b>SYSTEM ANALYSIS</b>	<b>4</b>
	2.1 Existing System	4
	2.2 Proposed System	4
	2.3 Advantages	4
	2.4 Modules and Functionalities	5
	2.5 Feasibility Study	6
	2.6 Conclusion	7
<b>3</b>	<b>SYSTEM DESIGN</b>	<b>8</b>
	3.1 System Design	8
	3.2 Software Design	9
	3.3 Input/Output Design	10

	3.4 Software Requirement Specifications (SRS)	10
	3.5 UML Concepts	11
	3.6 Hardware and Software Requirements	22
<b>4</b>	<b>IMPLEMENTATION AND CODING</b>	<b>23</b>
	4.1 Implementation	23
	4.2 Object Detection	23
	4.3 Extreme Learning Machine, CSS	24
	4.4 Python programming	26
	4.5 Haarcascade Features	26
	4.6 Coding	27
<b>5</b>	<b>TESTING</b>	<b>29</b>
	5.1 Software Testing	29
	5.2 Testing Design	30
	5.3 Test case1	31
<b>6</b>	<b>LIST OF IMAGES</b>	<b>33</b>
	6.1 India Table	33
	6.2 Future Data	34
	6.3 Litter/Garbage detection output	35
	6.4 Output of litter detection code	36
	6.5 Litter detection.png	37
<b>7</b>	<b>CONCLUSION AND FUTURE ENHANCEMENTS</b>	<b>38</b>
<b>8</b>	<b>REFERENCES</b>	<b>39</b>

## ABSTRACT

---

OpenCV is the huge and open-source library for image processing, machine learning and computer vision. It is also playing an important role in real-time operation. With the help of the OpenCV library, we can easily process the images as well as videos to identify the objects, faces or even handwriting of a human present in the file. We will only focus to object detection from images using OpenCV. We will learn about how we can use OpenCV to do object detection from a given image using a Python program. Object detection is a modern computer technology that is related to image processing, deep learning and computer vision to detect the objects present in an image file. All the technologies used in the Object detection technique (as we mentioned earlier) deals with detecting instances of the object in the image or video. We will first import the OpenCV library in the Python program, and then we will use functions to perform object detection on an image file given to us. But, before using and importing the library functions, let's first install the requirements for using the Object detection technique. We will detect the garbage detection based on civilization in Rural and Urban areas. We had created a dashboard where it shows graph about where the garbage is more by this it is easy to clear it. The requirement to perform object detection using the OpenCV library is that the OpenCV library should be present in our device so that we can import it into a Python program and use its object detection functions. If this library is not present in our system, we can use the following command in our command prompt terminal to install it. Matplotlib is very helpful in the opening, closing, reading etc., images in a Python program, and that's why the installation of this library for object detection becomes an important requirement. If the matplotlib library is not present in our system, we have to use the following command in our command prompt terminal to install it.

## **LIST OF FIGURES**

<b>FIG. NO</b>	<b>FIG. NAME</b>	<b>PAGE NO.</b>
3.a	System Architecture Design	9
3.b	Application Architecture Design	9
3.c	Class Diagram	5
3.d	Use Case Diagram	19
3.e	Activity Diagram	20
3.f	Deployment Diagram	21
5.a	Testcase-1	31
6.a	India Table	33
6.b	Future Data	34
6.c	Litter/Garbage detection output	35
6.d	Output of litter detection code	36
6.e	Litter detection.png	37



## SYMBOLS & ABBREVIATIONS

---

<b>HTML</b>	: Hyper Text Markup Language
<b>XML</b>	: Extensible Markup Language.
<b>Python</b>	: coding.
<b>OpenCV</b>	: Object detection.
<b>Matplotlib</b>	: Graphical representation.
<b>PNG</b>	: Portable Network Graphics.
<b>ImRead</b>	: Function used to open the image file.

# Chapter 1

## INTRODUCTION TO PROJECT

In this project we are going to detect litter on roads based on civilization. To implement this project we use OpenCV, Matplotlib packages for object detection and we will use the Haar cascade technique to do object detection. Basically, the Haar cascade technique is an approach based on machine learning where we use a lot of positive and negative images to train the classifier to classify between the images. Haar cascade classifiers are considered as the effective way to do object detection with the OpenCV library. Now, let's understand the concept of positive and negative images.

**Positive images:** These are the images that contain the objects which we want to be identified from the classifier.

**Negative Images:** These are the images that do not contain any object that we want to be detected by the classifier, and these can be images of everything else.

I aim to raise people's awareness and encourage them to recycle so that the oceans, seas, or the environment do not become more polluted and damaged in our increasingly polluted world. This project is aimed to detect recyclable objects such as cardboard, paper, plastic, and metal.

### 1.1 PURPOSE OF THE PROJECT

The purpose of this project is to help keep the cities clean. And it is also help in reducing the plastic wastage which affects the environment. Reducing solid waste is reducing the amount of trash that goes to landfills. This project shows the detail information regarding how much percentage of trash is present in an area using the number of people living. It also helps in clear the oceans, seas wastage by using garbage detector.

## **1.2 SCOPE OF THE PROJECT**

Smart Civilization for Detecting Littering on Roads Project is mostly used to reduce the garbage on roads which helps people regarding health and helps in saving environment. Major scope is to reduce the garbage and helps in good environment.

## **1.3 FEATURES OF THE PROJECT**

Object detection, which not only requires accurate classification of objects in images but also needs accurate location of objects is an automatic image detection process based on statistical and geometric features. The accuracy of object classification and object location is important indicators to measure the effectiveness of model detection. Object detection is widely used in intelligent monitoring, military object detection, UAV navigation, unmanned vehicle, and intelligent transportation. However, because of the diversity of the detected objects, the current model fails to detect objects. The changeable light and the complex background increase the difficulty of the object detection especially for the objects that are in the complex environment. The traditional method of image classification and location by multiscale pyramid method needs to extract the statistical features of the image in multiscale and then classify the image by a classifier. Because different types of images are characterized by different features, it is difficult to use one or more features to represent objects, which do not achieve a robust classification model. Those models failed to detect the objects especially that there are more detected objects in an image.

- Identifies the trash items by video capture.
- Use python packages like OpenCV and Matplotlib.
- Follows the Haar Cascade Technique.
- Implementing code for detecting images.
- Use of detect Multiscale function.
- Creating an xml file for cascade the classifier.
- Giving labels to the image like paper, metal, plastic etc.

## **1.4 PROBLEM DEFINITION**

The threat of waste to the environment, health and safety is huge. And so are the financial and social ramifications, waste experts say. Pollution runs into rivers and seeps into ground water. Flooding is caused by garbage clogging drains, and the atmosphere can be poisoned by the toxic discharge from trash. In normal garbage detection system, we only detect the litter by using datasets or video capture to know the type of the item like plastic, paper etc. But in proposed system we used an dashboard using power bi where it is used to predict the percentage of garbage on roads in form of graph where it helps to reduce the garbage on roads by knowing how much level of garbage on roads. It shows the difference in rural and urban. It gives information regarding population.

Problems arising In the existing system:

1. It can only detect the garbage on roads?
2. How can we find the percentage of garbage and give the graphical representation?

SOLUTION: The solution we created a dashboard where we can find the graphical representation in dashboard and shows the percentage of garbage (or) Litter on roads in a particular area.

## **1.5 ORGANIZATION OF DOCUMENT**

In this project documentation we have initially put the definition and objective of the project as well as the design of the project which is followed by the implementation and testing phases. The project has been concluded successfully and the future enhancements of the project also given in this documentation.

## **Chapter 2**

### **SYSTEM ANALYSIS**

#### **2.1 EXISTING SYSTEM:**

In Existing system, we only detect the litter by using datasets or video capture to know the type of the item like plastic, paper etc.

#### **2.2 PROPOSED SYSTEM:**

In this proposed system, we not only detect the litter by using datasets or video capture to know the type of the item like plastic, paper etc. But we also created a dashboard which helps in knowing the percentage of garbage in an area. This helps in knowing how negligence the people in that area.

#### **2.3 ADVANTAGES**

This proposed system has following advantages:

- Helps in knowing the area which has more litter on roads and taking necessity upon them.
- Knowing the percentage of litter.

#### **2.4 MODULES AND FUNCTIONALITIES**

Modules are:

- 1) imread
- 2) detect Multiscale
- 3) OpenCV and Matplotlib
- 4) imshow

## 5) Cascade Classifier

### 2.4.1 imread Function in python

The imread function is used to open the image file (PNG) in cv2. The image which is used for detecting must be opened that is done by imread () function.

### 2.4.2 Detect Multiscale Function

We use the detect Multiscale () function with the imported cascade file to detect the object present in the image or not.

### 2.4.3 OpenCV and Matplotlib

OpenCV library should be present in our device so that we can import it into a Python program and use its object detection functions. Matplotlib is very helpful in the opening, closing, reading etc., images in a Python program.

### 2.4.4 Imshow Function

It is used to display the processed image using the plt show () and imshow () function. This function is a part of Matplotlib library. This function helps in visualize the output screen.

### 2.4.5 Cascade Classifier

Cascading classifiers are trained with several hundred "positive" sample views of a particular object and arbitrary "negative" images of the same size. After the classifier is trained it can be applied to a region of an image and detect the object in q Cascade classifiers are available in OpenCV, with pre-trained cascades for frontal faces and upper body. Training a new cascade in OpenCV is also possible with either haartraining or train cascades methods. This can be used for rapid object detection of more specific targets, including non-human objects with Haar-like features question.

## **2.5 FEASIBILITY STUDY:**

---

The next step in analysis is to verify the feasibility of the proposed system. “All projects are feasible given unlimited resources and infinite time“. But in reality both resources and time are scarce. Project should confirm to time bound and should be optimal in their consumption of resources. This place a constant is approval of any project.

2.5.1 Technical feasibility

2.5.2 Operational feasibility

2.5.3 Economic feasibility

### **2.5.1 TECHNICAL FEASIBILITY**

To determine whether the proposed system is technically feasible, we should take into consideration the technical issues involved behind the system. This Application uses the web technologies, which is rampantly employed these days worldwide. The world without the web is incomprehensible today. That goes to proposed system is technically feasible.

### **2.5.2 OPERATIONAL FEASIBILITY:**

To determine the operational feasibility of the system we should take into consideration the awareness level of the users. This system is operational feasible since the users are familiar with the technologies and hence there is no need to gear up the personnel to use system. Also the system is very friendly and to use.

### **2.5.3 ECONOMIC FEASIBILITY:**

To decide whether a project is economically feasible, we have to consider various

Factors as:

- Cost benefit analysis
- Long-term returns
- Maintenance costs

It requires average computing capabilities and access to internet, which are very basic requirements and can be afforded by any organization hence it doesn't incur additional economic overheads, which renders the system economically feasible.

## **2.6 CONCLUSION**

In this phase, we understand the modules and functionalities of the system. We arrange all the required components to develop the project in this phase itself so that we will have a clear idea regarding the requirements before designing the project. Thus, we will proceed to the design phase followed by the implementation phase of the project.



## Chapter 3

### SYSTEM DESIGN

#### 3.1 SYSTEM DESIGN:

System design is transition from a user oriented document to programmers or data base personnel. The design is a solution, how to approach to the creation of a new system. This is composed of several steps. It provides the understanding and procedural details necessary for implementing the system recommended in the feasibility study. Designing goes through logical and physical stages of development, logical design reviews the present physical system, prepare input and output specification, details of implementation plan and prepare a logical design walkthrough.

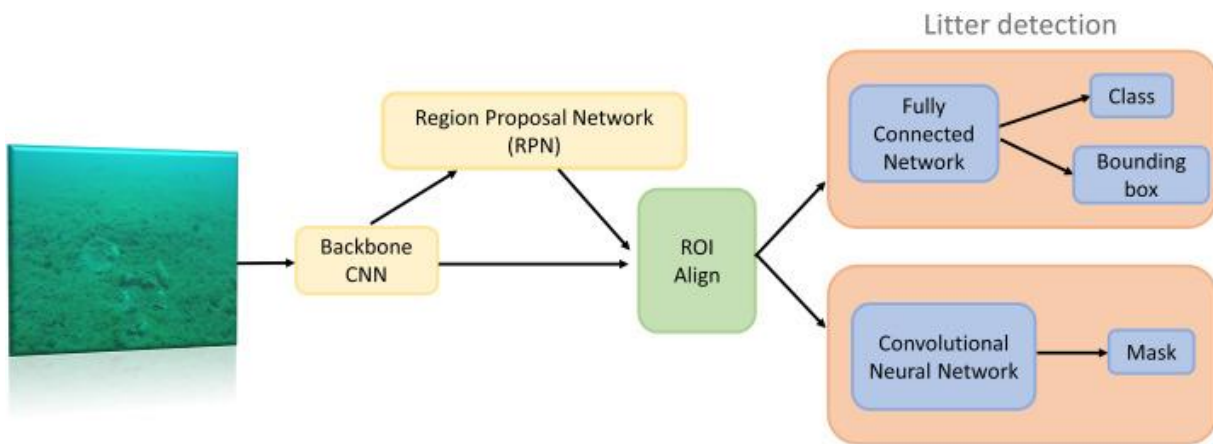
The database tables are designed by analyzing functions involved in the system and format of the fields is also designed. The fields in the database tables should define their role in the system. The unnecessary fields should be avoided because it affects the storage areas of the system. Then in the input and output screen design, the design should be made user friendly. The menu should be precise and compact.

#### 3.2 SOFTWARE DESIGN

In designing the software following principles are followed:

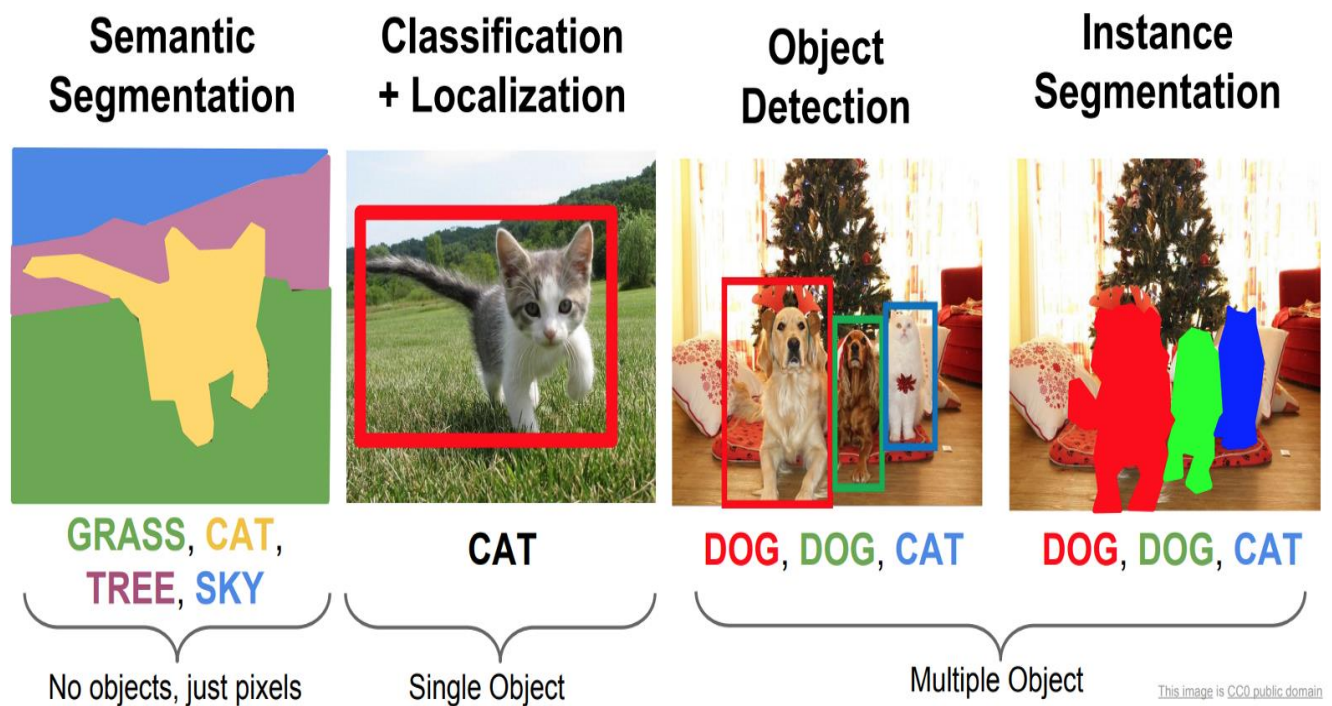
- ❖ **Modularity and partitioning:** software is designed such that, each system should consists of hierarchy of modules and serve to partition into separate function.
- ❖ **Coupling:** modules should have little dependence on other modules of a system.
- ❖ **Cohesion:** modules should carry out in a single processing function.
- ❖ **Shared use:** avoid duplication by allowing a single module be called by other that need the function it provides

### 3.2.1 SYSTEM ARCHITECTURE DESIGN



**Fig:3.a System Architecture Design**

### APPLICATION ARCHITECTURE DESIGN



**Fig:3.b Application Architecture Design**

### **3.3 INPUT/OUTPUT DESIGN**

#### **3.3.1 Input design:**

Considering the requirements, procedures to collect the necessary input data in most efficiently designed. The input design has been done keeping in view that, the interaction of the user with the system being the most effective and simplified way.

Also the measures are taken for the following

- Avoid unauthorized access
- Eliminating extra steps
- Keeping the process simple
- At this stage the sender and receiver action is designed

#### **3.3.2 Output design:**

All the screens of the system are designed with a view to provide the user with easy operations in simpler and efficient way, minimum key strokes possible. Instructions and important information is emphasized on the screen. Almost every screen is provided with no error and important messages and option selection facilitates. Emphasis is given for speedy processing and speedy transaction between the screens. Each screen assigned to make it as much user friendly as possible by using interactive procedures. So to say user can operate the system without much help from the operating manual.

### **3.4 SOFTWARE REQUIREMENT SPECIFICATIONS**

#### **3.4.1 What is SRS?**

Software Requirement Specification (SRS) is the starting point of the software developing activity. As system grew more complex it became evident that the goal of the entire system cannot be easily comprehended. Hence the need for the requirement phase arose. The software project is initiated by the client needs. The SRS is the means of translating the ideas of the minds of clients (the input) into a formal document (the output of the requirement phase.) The SRS phase consists of two basic activities:

### **3.4.2 Problem/Requirement Analysis:**

The process is order and more nebulous of the two, deals with understand the problem, the goal and constraints.

### **3.4.3 Requirement Specification:**

Here, the focus is on specifying what has been found giving analysis such as representation, specification languages and tools, and checking the specifications are addressed during this activity.

The Requirement phase terminates with the production of the validate SRS document. Producing the SRS document is the basic goal of this phase.

### **3.4.4 Document Conventions:**

We have used Times New Roman (text size 12).Bold Font is used for Main Headings (text size of 16). Normal font is used for sub headings (text size of 14).

**Font:** Times New Roman

**Main Heading:** Bold Font

## **3.5 UML Concepts:**

The Unified Modeling Language (UML) is a standard language for writing software blue prints. The UML is a language for

- Visualizing
- Specifying
- Constructing
- Documenting the artifacts of a software intensive system.

The UML is a language which provides vocabulary and the rules for combining words in that vocabulary for the purpose of communication. A modeling language is a language whose vocabulary and the rules focus on the conceptual and physical representation of a system. Modeling yields an understanding of a system.

### **3.5.1 Building Blocks of the UML:**

The vocabulary of the UML encompasses three kinds of building blocks:

- Things
- Relationships
- Diagrams

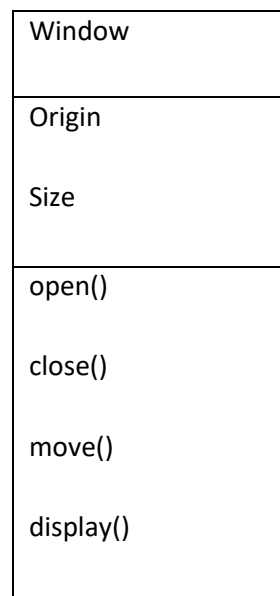
Things are the abstractions that are first-class citizens in a model; relationships tie these things together; diagrams group interesting collections of things.

### 3.5.2 Things in the UML:

There are four kinds of things in the UML:

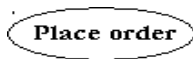
- Structural things
- Behavioral things
- Grouping things
- Annotational things

❖ **Structural things** are the nouns of UML models. The structural things used in the project design are first, a **class** is a description of a set of objects that share the same attributes, operations, relationships and semantics.



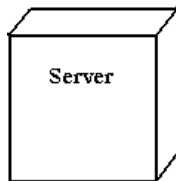
**Fig 3.c: Classes**

Second, a use case is a description of set of sequence of actions that a system performs that yields an observable result of value to particular actor



## Use Cases

Third, a node is a physical element that exists at runtime and represents a computational resource, generally having at least some memory and often processing capability

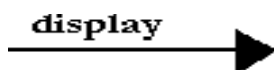


## Nodes

Behavioral things are the dynamic parts of UML models. The behavioral thing used is:

### Interaction:

An interaction is a behavior that comprises a set of messages exchanged among a set of objects within a particular context to accomplish a specific purpose. An interaction involves a number of other elements, including messages, action sequences (the behavior invoked by a message, and links (the connection between objects).



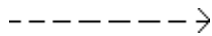
### Messages:

#### 3.5.2 Relationships in the UML:

There are four kinds of relationships in the UML:

- Dependency
- Association
- Generalization
- Realization

A **dependency** is a semantic relationship between two things in which a change to one thing may affect the semantics of the other thing (the dependent thing).



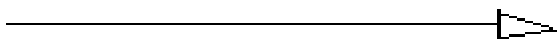
## Dependencies

An association is a structural relationship that describes a set of links, a link being a connection among objects. Aggregation is a special kind of association, representing a structural relationship between a whole and its parts.



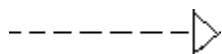
## Association

A generalization is a specialization/ generalization relationship in which objects of the specialized element (the child) are substitutable for objects of the generalized element (the parent).



## Generalization

A realization is a semantic relationship between classifiers, where one classifier specifies a contract that another classifier guarantees to carry out.



## Realization

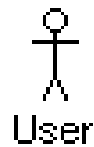
### 3.5.3 Sequence Diagrams:

UML sequence diagrams are used to represent the flow of messages, events and actions between the objects or components of a system. Time is represented in the vertical direction showing the sequence of interactions of the header elements, which are displayed horizontally at the top of the diagram.

Sequence Diagrams are used primarily to design, document and validate the architecture, interfaces and logic of the system by describing the sequence of actions that need to be performed to complete a task or scenario. UML sequence diagrams are useful design tools because they provide a dynamic view of the system behavior which can be difficult to extract from static diagrams or specifications.

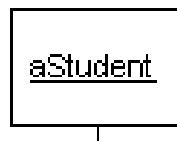
### **Actor**

Represents an external person or entity that interacts with the system



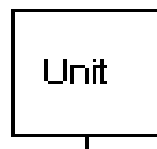
### **Object**

Represents an object in the system or one of its components



### **Unit**

Represents a subsystem, component, unit, or other logical entity in the system (may or may not be implemented by objects)



### ❖ **Separator**

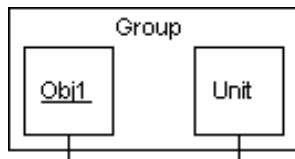
Represents an interface or boundary between subsystems, components or units (e.g., air interface, Internet, network)



### ❖ **Group**

Groups related header elements into subsystems or components





## Sequence Diagram Body Elements

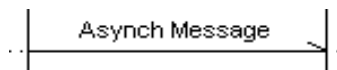
### ❖ Action

Represents an action taken by an actor, object or unit



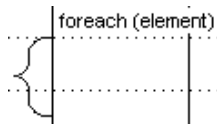
### ❖ Asynchronous Message

An asynchronous message between header elements



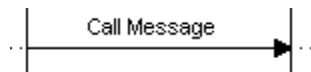
### ❖ Block

A block representing a loop or conditional for a particular header element



### ❖ Call Message

A call (procedure) message between header elements



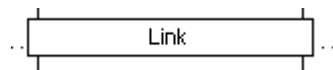
### ❖ Create Message

A "create" message that creates a header element (represented by lifeline going from dashed to solid pattern)

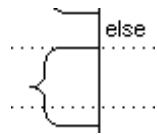


### ❖ Diagram Link

Represents a portion of a diagram being treated as a functional block. Similar to a procedure or function call that abstracts functionality or details not shown at this level. Can optionally be linked to another diagram for elaboration.



Else Block Represents an "else" block portion of a diagram block



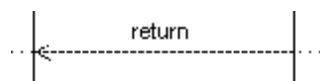
#### ❖ Message

A simple message between header elements



#### ❖ Return Message

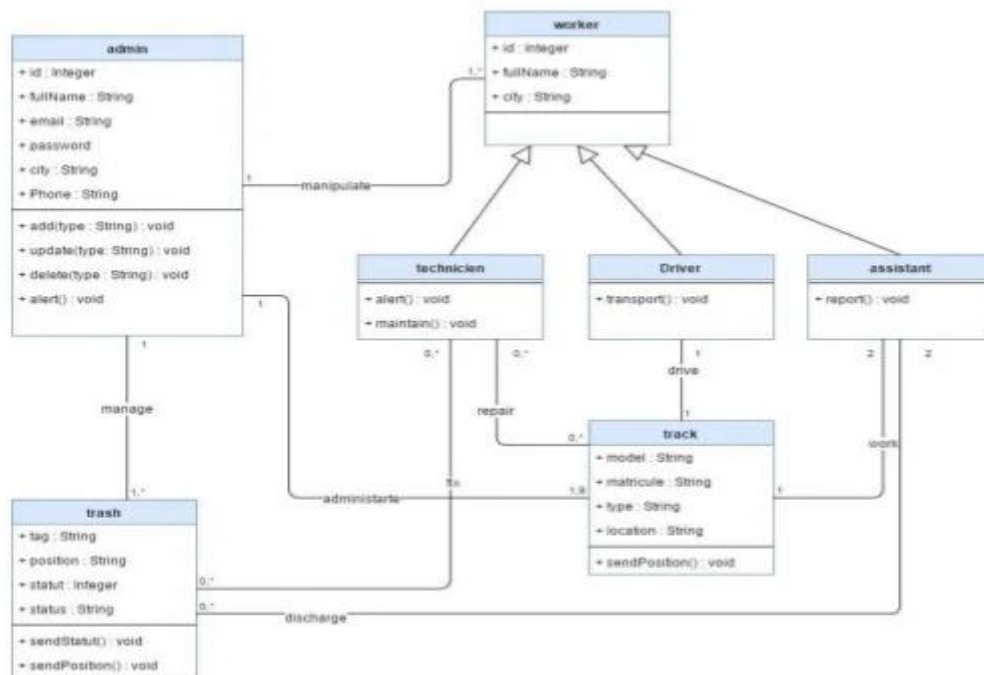
A return message between header elements



## Class Diagram:

A class diagram in the Unified Modelling Language is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among objects. Class diagram is a static diagram. It represents the static view of an application. Class diagram describes the attributes and operations of a class and the constraints imposed on the system.

Class diagram:

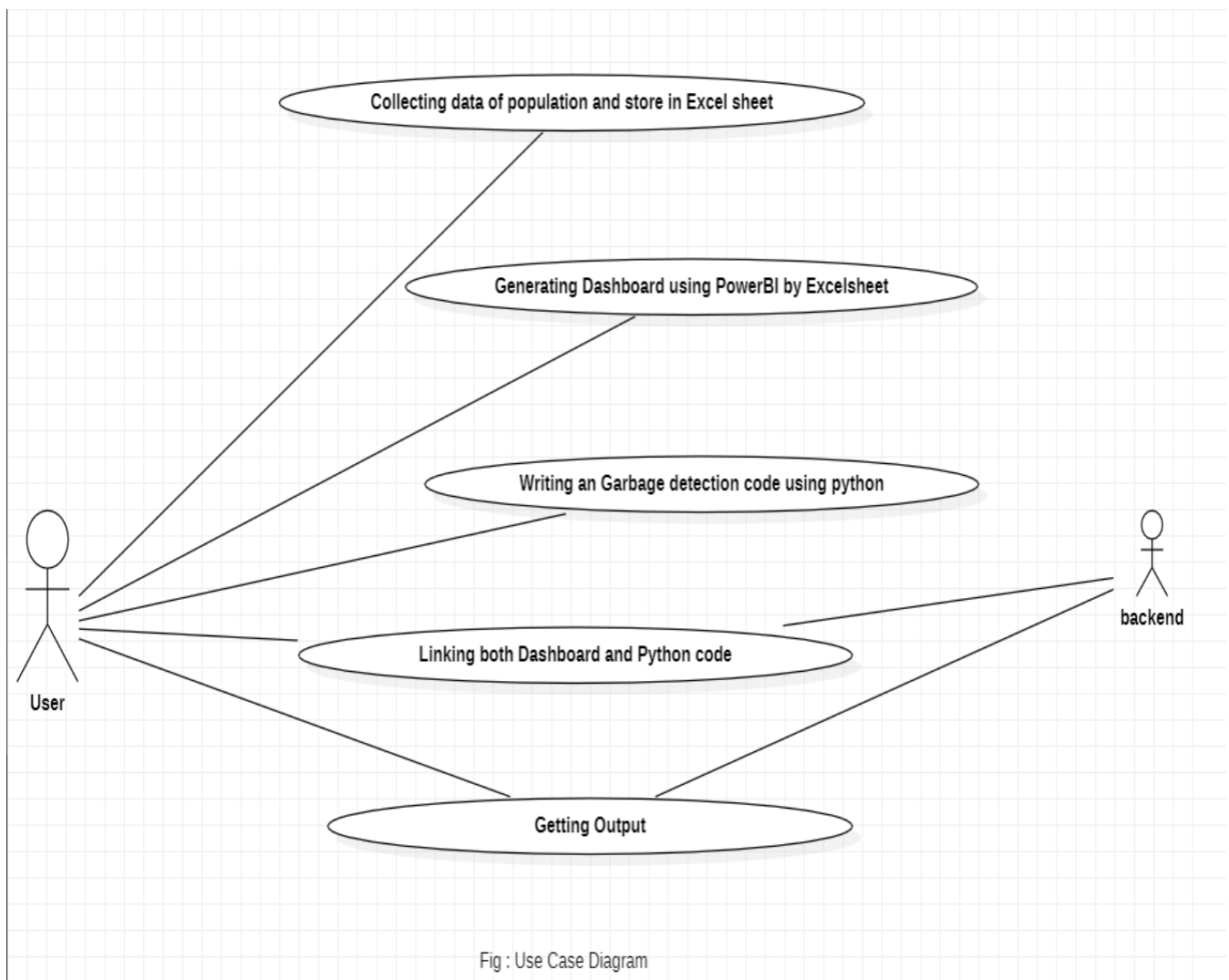


class diagram

Fig 3.c: Class Diagram

## Use Case Diagram:

A use case diagram at its simplest is a representation of a user's interaction with the system that shows the relationship between the user and the different use cases in which the user is involved. A use case diagram can identify the different types of users of a system and the different use cases and will often be accompanied by other types of diagrams as well. The use cases are represented by either circles or ellipses.

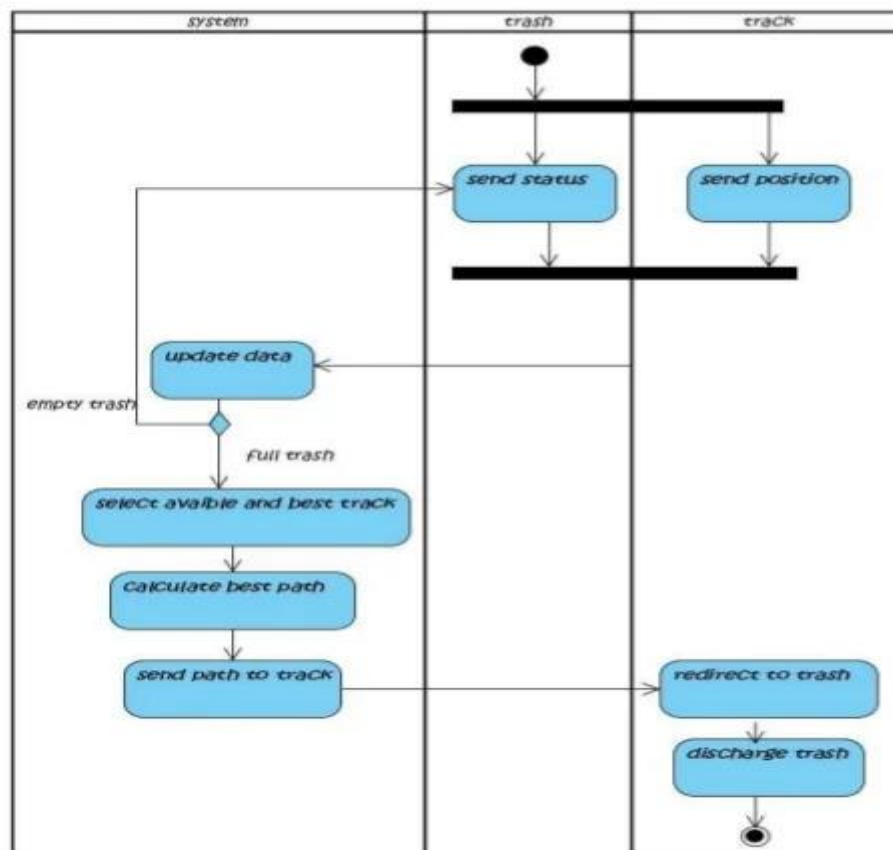


**Fig 3.d: Use Case Diagram**

## 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 are intended to model both computational and organizational processes (i.e., workflows) as well as the data flows intersecting with the related activities.

Activity diagram:



activity diagram

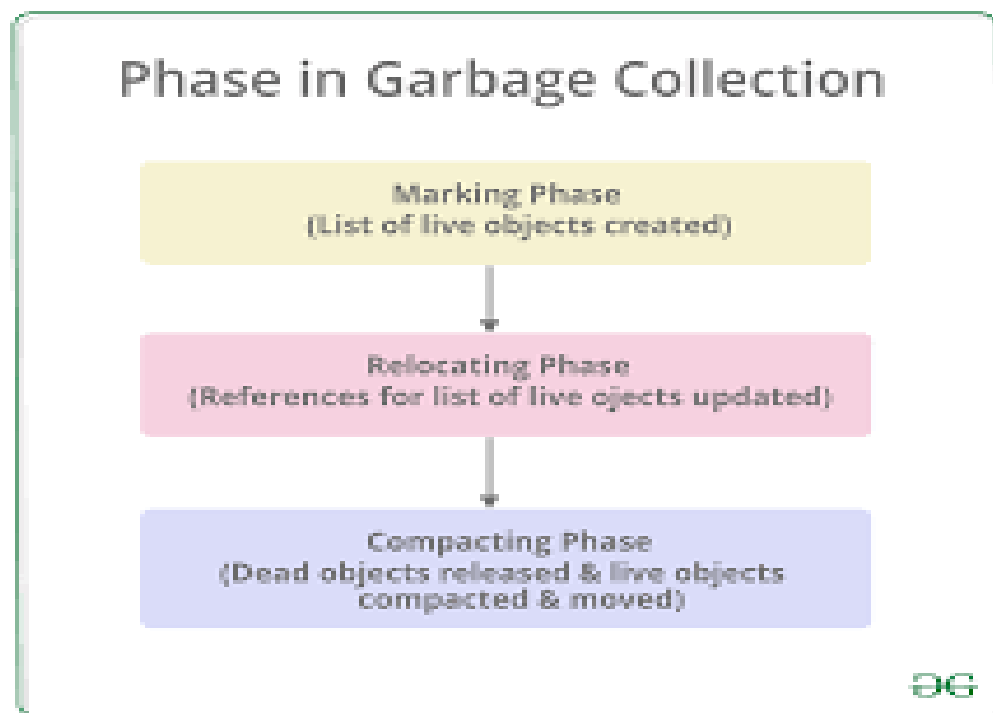
Fig 3.e: Activity Diagram

## Phases in Garbage collection:

**Marking Phase:** A list of all the live objects is created during the marking phase. This is done by following the references from all the root objects. All of the objects that are not on the list of live objects are potentially deleted from the heap memory.

**Relocating Phase:** The references of all the objects that were on the list of all the live objects are updated in the relocating phase so that they point to the new location where the objects will be relocated to in the compacting phase.

**Compacting Phase:** The heap gets compacted in the compacting phase as the space occupied by the dead objects is released and the live objects remaining are moved. All the live objects that remain after the garbage collection are moved towards the older end of the heap memory in their original order.



**Fig 3.f : Phases in Garbage detection**

## **3.6 HARDWARE AND SOFTWARE REQUIREMENTS:**

### **3.6.1 Software Interfaces:**

We use Python IDE and PyCharm for executing Python code which we have writing for Garbage Detection. We collect Data of past and Future and store in Excel. And Creating a Dashboard using PowerBI for Detail information of garbage present in an area.

### **3.6.2 Software Requirements**

- Python PyCharm developer edition.
- OpenCV and Matplotlib libraries.
- Python IDE.
- Operating System - Windows XP/7/8/10.
- PowerBI Application.
- MS Excel.

### **3.6.3 Hardware Requirements**

- Ram: : 1GB Ram and above
- Hard Disk : 50GB and above
- Processor : Dual core and above

## **Chapter 4**

### **IMPLEMENTATION AND CODING**

#### **4.1 IMPLEMENTATION**

##### **4.2 Object Detection**

###### **4.2.1 OpenCV:**

OpenCV (Opensource Computer Vision Library) is an open source computer vision and machine learning software library. OpenCV was built to provide a common infrastructure for computer vision applications and to accelerate the use of machine perception in the commercial products. Being a BSD-licensed product, OpenCV makes it easy for businesses to utilize and modify the code.

###### **4.2.1.1 Matplotlib:**

Matplotlib is a plotting library for the Python programming language and its numerical mathematics extension NumPy. It provides an object-oriented API for embedding plots into applications using general-purpose GUI toolkits like Tkinter, wxPython, Qt, or GTK. There is also a procedural "pylab" interface based on a state machine (like OpenGL), designed to closely resemble that of MATLAB, though its use is discouraged. SciPy makes use of Matplotlib. Matplotlib was originally written by "John D. Hunter".

###### **4.2.1.2 Video capture using OpenCV:**

Python provides various libraries for image and video processing. One of them is OpenCV. OpenCV is a vast library that helps in providing various functions for image and video operations. With OpenCV, we can capture a video from the camera. It lets you create a video capture object which is helpful to capture videos through webcam and then you may perform desired operations on that video.

###### **4.2.1.3 Steps to Capture a video:**

- Use `cv2.VideoCapture()` to get a video capture object for the camera.
- Set up an infinite while loop and use the `read ()` method to read the frames using the above created object.
- Use `cv2.imshow()` method to show the frames in the video.
- Breaks the loop when the user clicks a specific key.



#### 4.2.1.4 wait key () function:

wait key () function of Python OpenCV allows users to display a window for given milliseconds or until any key is pressed. It takes time in milliseconds as a parameter and waits for the given time to destroy the window, if 0 is passed in the argument it waits till any key is pressed.

**Syntax:** cv2.waitKey(1)

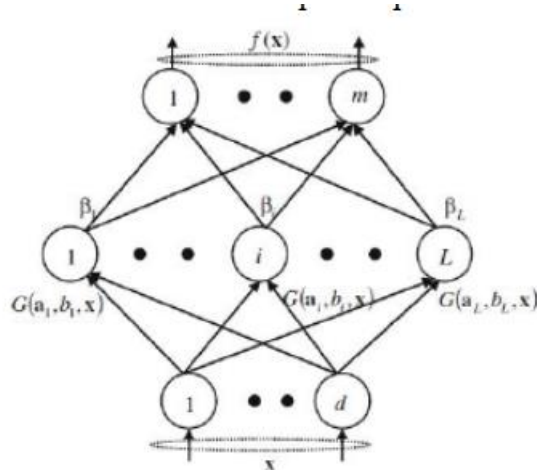
#### 4.2.1.5 Result

It gives the detection output and get detail analysis in dashboard.

### 4.3 Extreme Learning Machine, CSS, Haar Cascade Features:

#### 4.3.1 Extreme Learning Machine

Extreme Learning Machine (ELM) is a single-layer forward feedback neural network proposed by “Huang et al”. The input weights and hidden layer biases are all generated by program randomization instead of setting artificially. The output weights are determined by least squares regression. Therefore, the whole training process only needs to specify the number of hidden layer neurons and the activation function. Compared with the traditional neural network algorithm, the training speed is faster and the parameter selection is more flexible. At present, ELM has demonstrated its superior performance in many fields,



**Fig.2.** Single-layer ELM structure. ( $m=2$ ,  $L=20$ ,  $d=28$ ).

### 4.3.2 detect Multiscale Function:

detect Multiscale function is used to detect the faces. This function will return a rectangle with coordinates(x,y,w,h) around the detected face.

It takes 3 common arguments — the input image, scale Factor, and minNeighbours.

**scale Factor** specifies how much the image size is reduced with each scale. In a group photo, there may be some faces which are near the camera than others. Naturally, such faces would appear more prominent than the ones behind. This factor compensates for that.

**minNeighbours** specifies how many neighbours each candidate rectangle should have to retain it. You can read about it in detail here. You may have to tweak these values to get the best results. This parameter specifies the number of neighbours a rectangle should have to be called a face. We obtain these values after trial and test over a specific range.

### 4.3.3 Haar Cascade:

Haar-like feature is a simple rectangular feature introduced in the face detection system by Viola et al. and named after the Haar wavelet. Lienhart R et al. extended it further by adding rectangular features with a rotation of  $45^\circ$ . The extended features are roughly divided into three types: edge features, line features, center-surround features (as shown in Fig. 1). The Haar-like feature can effectively reflect the local gray change information of the image, and can also be quickly calculated through the integral image. In this paper, we select linear features and center-surround features as templates. The linear features Fig.1 (2a) -(2b) are computed in the dimensions of  $2 \times 3$  and  $2 \times 4$ , respectively. They are rotated at  $45^\circ$  and  $90^\circ$  to get new features Fig.1 (2c) -(2h). The feature Fig.1 (3a) uses a  $3 \times 3$  window, which is also rotated through  $45^\circ$  to get feature Fig.1 (3b). Finally, it is calculated using the integral image and 10 feature values are extracted for each pixel.

#### 4.3.3.1 Haar Cascade features:

##### 1. Edge features



##### 2. Line features



##### 3. Center-surround features



## 4.4 Python programming:

Python is an interpreted high-level general-purpose programming language. Its design philosophy emphasizes code readability with its use of significant indentation. Its language constructs as well as its object-oriented approach aim to help programmers write clear, logical code for small and large-scale projects. Python is dynamically-typed and garbage-collected. It supports multiple programming paradigms, including structured (particularly, procedural), object-oriented and functional programming. It is often described as a "batteries included" language due to its comprehensive standard library. Guido van Rossum began working on Python in the late 1980s, as a successor to the ABC programming language, and first released it in 1991 as Python 0.9.0.[33] Python 2.0 was released in 2000 and introduced new features, such as list comprehensions and a garbage collection system using reference counting. Python 3.0 was released in 2008 and was a major revision of the language that is not completely backward-compatible. Python 2 was discontinued with version 2.7.18 in 2020. Python consistently ranks as one of the most popular programming languages.

### 4.4.1 : cv2.imread() method

OpenCV-Python is a library of Python bindings designed to solve computer vision problems. cv2.imread() method loads an image from the specified file. If the image cannot be read (because of missing file, improper permissions, unsupported or invalid format) then this method returns an empty matrix.

**Syntax:** cv2.imread(path, flag)

#### **Parameters:**

**path:** A string representing the path of the image to be read.

**flag:** It specifies the way in which image should be read. It's default value is cv2.IMREAD\_COLOR

**Return Value:** This method returns an image that is loaded from the specified file.

#### **Types of flags:**

**cv2.IMREAD\_COLOR:** It specifies to load a color image. Any transparency of image will be neglected. It is the default flag. Alternatively, we can pass integer value 1 for this flag.

**cv2.IMREAD\_GRAYSCALE:** It specifies to load an image in grayscale mode. Alternatively, we can pass integer value 0 for this flag.

**cv2.IMREAD\_UNCHANGED:** It specifies to load an image as such including alpha channel. Alternatively, we can pass integer value -1 for this flag.

## 4.6 CODING:

### 4.4.1 Garbage.xml File

```
<?xml version="1.0"?>
<opencv_storage>
<cascade>
  <stageType>BOOST</stageType>
  <featureType>HAAR</featureType>
  <height>24</height>
  <width>24</width>
  <stageParams>
    <boostType>GAB</boostType>
    <minHitRate>9.9500000476837158e-01</minHitRate>
    <maxFalseAlarm>5.0000000000000000e-01</maxFalseAlarm>
    <weightTrimRate>9.4999998807907104e-01</weightTrimRate>
    <maxDepth>1</maxDepth>
    <maxWeakCount>100</maxWeakCount></stageParams>
  <featureParams>
    <maxCatCount>0</maxCatCount>
    <featSize>1</featSize>
    <mode>BASIC</mode></featureParams>
  <stageNum>15</stageNum>
  <stages>
    <!-- stage 0 -->
    <_>
      <maxWeakCount>7</maxWeakCount>
      <stageThreshold>-6.6240966320037842e-01</stageThreshold>
      <weakClassifiers>
        <_>
          <internalNodes>
```

```

    0 -1 85 1.2447724584490061e-03</internalNodes>
    <leafValues>
    -4.5525291562080383e-01 6.1764705181121826e-01</leafValues></_>
<_>
    <internalNodes>
    0 -1 58 -1.0394077980890870e-03</internalNodes>
    <leafValues>
    7.2507286071777344e-01 -2.4769568443298340e-01</leafValues></_>
<_>
    <internalNodes>
    0 -1 74 1.7180796712636948e-02</internalNodes>
    <leafValues>
    -2.1392868459224701e-01 7.3991668224334717e-01</leafValues></_>
<_>
    <internalNodes>
    0 -1 6 3.6049410700798035e-02</internalNodes>
    <leafValues>
    2.0645608007907867e-01 -9.0167343616485596e-01</leafValues></_>
<_>
    <internalNodes>
    0 -1 43 -1.7439091205596924e-01</internalNodes>
    <leafValues>
    -9.1454881429672241e-01 2.4217011034488678e-01</leafValues></_>
<_>
    <internalNodes>
    0 -1 41 -5.1970762433484197e-04</internalNodes>
    <leafValues>
    6.4517176151275635e-01 -3.4610882401466370e-01</leafValues></_>
<_>
    <internalNodes>
    0 -1 120 3.2015414035413414e-05</internalNodes>
    <leafValues>
    3.0623441934585571e-01 -6.0653960704803467e-
01</leafValues></_></weakClassifiers></_>
<!-- stage 1 -->
<_>
    <maxWeakCount>5</maxWeakCount>

```

```

<stageThreshold>-5.5276966094970703e-01</stageThreshold>
<weakClassifiers>
  <_>
    <internalNodes>
      0 -1 65 -2.3404557723551989e-03</internalNodes>
    <leafValues>
      8.6666667461395264e-01 -4.0714284777641296e-01</leafValues></_>
  <_>
    <internalNodes>
      0 -1 83 1.6290082130581141e-03</internalNodes>
    <leafValues>
      -2.3489895462989807e-01 7.7064388990402222e-01</leafValues></_>
  <_>
    <internalNodes>
      0 -1 106 8.5107320919632912e-03</internalNodes>
    <leafValues>
      -2.4215476214885712e-01 7.6241898536682129e-01</leafValues></_>
  <_>
    <internalNodes>
      0 -1 103 -5.6269816122949123e-03</internalNodes>
    <leafValues>
      7.974316477755737e-01 -2.6811656355857849e-01</leafValues></_>
  <_>
    <internalNodes>
      0 -1 57 -1.5520547749474645e-03</internalNodes>
    <leafValues>
      5.9954345226287842e-01 -3.6586087942123413e-
01</leafValues></_></weakClassifiers></_>
<!-- stage 2 -->
<_>
  <maxWeakCount>7</maxWeakCount>
  <stageThreshold>-1.2709754705429077e+00</stageThreshold>
  <weakClassifiers>
    <_>
      <internalNodes>
        0 -1 50 1.8419425934553146e-02</internalNodes>
      <leafValues>
        -4.0501791238784790e-01 8.2608693838119507e-01</leafValues></_>
    <_>
      <internalNodes>
        0 -1 76 3.9332383312284946e-03</internalNodes>
      <leafValues>
        -2.4419558048248291e-01 7.4375104904174805e-01</leafValues></_>

```

```

<_>
  <internalNodes>
    0 -1 93 7.5550392270088196e-02</internalNodes>
  <leafValues>
    -2.4231182038784027e-01 7.5593370199203491e-01</leafValues></_>
<_>
  <internalNodes>
    0 -1 92 -3.7791917566210032e-03</internalNodes>
  <leafValues>
    6.7227303981781006e-01 -3.2257071137428284e-01</leafValues></_>
<_>
  <internalNodes>
    0 -1 86 -7.1782581508159637e-03</internalNodes>
  <leafValues>
    7.1929007768630981e-01 -2.1317864954471588e-01</leafValues></_>
<_>
  <internalNodes>
    0 -1 114 2.5790577637962997e-04</internalNodes>
  <leafValues>
    3.1687000393867493e-01 -6.0512220859527588e-01</leafValues></_>
<_>
  <internalNodes>
    0 -1 19 1.1163683608174324e-02</internalNodes>
  <leafValues>
    -2.3342229425907135e-01 8.0238491296768188e-
01</leafValues></_></weakClassifiers></_>
<!-- stage 3 -->
<_>
  <maxWeakCount>6</maxWeakCount>
  <stageThreshold>-8.0608248710632324e-01</stageThreshold>
  <weakClassifiers>
    <_>
      <internalNodes>
        0 -1 28 -1.5377387404441833e-02</internalNodes>
      <leafValues>
        6.8965518474578857e-01 -4.3071159720420837e-01</leafValues></_>
    <_>
      <internalNodes>
        0 -1 29 1.6768474131822586e-02</internalNodes>
      <leafValues>
        -2.6171728968620300e-01 6.8303465843200684e-01</leafValues></_>
    <_>
      <internalNodes>

```

```

    0 -1 30 -1.0023083537817001e-02</internalNodes>
    <leafValues>
    8.7276935577392578e-01 -1.6868470609188080e-01</leafValues></_>
  <_>
    <internalNodes>
    0 -1 73 -4.5953881926834583e-03</internalNodes>
    <leafValues>
    6.9845098257064819e-01 -2.6742115616798401e-01</leafValues></_>
  <_>
    <internalNodes>
    0 -1 82 3.1605376861989498e-03</internalNodes>
    <leafValues>
    -2.2918422520160675e-01 5.8094346523284912e-01</leafValues></_>
  <_>
    <internalNodes>
    0 -1 17 -1.5070034191012383e-02</internalNodes>
    <leafValues>
    6.5976381301879883e-01 -2.5849115848541260e-
01</leafValues></_></weakClassifiers></_>
  <!-- stage 4 -->
  <_>
    <maxWeakCount>9</maxWeakCount>
    <stageThreshold>-1.2865171432495117e+00</stageThreshold>
    <weakClassifiers>
    <_>
      <internalNodes>
      0 -1 63 -2.1933926269412041e-02</internalNodes>
      <leafValues>
      6.2962961196899414e-01 -4.0221402049064636e-01</leafValues></_>
    <_>
      <internalNodes>
      0 -1 94 4.9441345036029816e-03</internalNodes>
      <leafValues>
      -2.7928054332733154e-01 5.2863395214080811e-01</leafValues></_>
    <_>
      <internalNodes>
      0 -1 90 1.3970070540381130e-05</internalNodes>
      <leafValues>
      -2.9882851243019104e-01 4.8778983950614929e-01</leafValues></_>
    <_>
      <internalNodes>
      0 -1 51 -1.1160170288349036e-05</internalNodes>
      <leafValues>

```



```

    4.7322851419448853e-01 -3.0657032132148743e-01</leafValues></_>
<_>
  <internalNodes>
    0 -1 102 -1.0583555325865746e-02</internalNodes>
  <leafValues>
    8.1141984462738037e-01 -1.9614291191101074e-01</leafValues></_>
<_>
  <internalNodes>
    0 -1 81 -1.0073794983327389e-03</internalNodes>
  <leafValues>
    -4.9484735727310181e-01 3.2896992564201355e-01</leafValues></_>
<_>
  <internalNodes>
    0 -1 38 2.0592916756868362e-02</internalNodes>
  <leafValues>
    -2.3296265304088593e-01 7.8816026449203491e-01</leafValues></_>
<_>
  <internalNodes>
    0 -1 53 1.8912513041868806e-05</internalNodes>
  <leafValues>
    -3.0465725064277649e-01 5.1173448562622070e-01</leafValues></_>
<_>
  <internalNodes>
    0 -1 79 4.6662453678436577e-04</internalNodes>
  <leafValues>
    -3.8144919276237488e-01 4.4471690058708191e-
01</leafValues></_></weakClassifiers></_>
<!-- stage 5 -->
<_>
  <maxWeakCount>8</maxWeakCount>
  <stageThreshold>-8.6118948459625244e-01</stageThreshold>
  <weakClassifiers>
    <_>
      <internalNodes>
        0 -1 22 -1.4273080043494701e-02</internalNodes>
      <leafValues>
        5.3125000000000000e-01 -4.1762453317642212e-01</leafValues></_>
    <_>
      <internalNodes>
        0 -1 118 2.5774464011192322e-03</internalNodes>
      <leafValues>
        2.8723919391632080e-01 -5.1181358098983765e-01</leafValues></_>
    <_>

```

```

    <internalNodes>
      0 -1 44 1.1160170288349036e-05</internalNodes>
    <leafValues>
      -3.6608049273490906e-01 3.9194977283477783e-01</leafValues></_>
  <_>
    <internalNodes>
      0 -1 26 3.3826173748821020e-03</internalNodes>
    <leafValues>
      -1.6742253303527832e-01 8.4731590747833252e-01</leafValues></_>
  <_>
    <internalNodes>
      0 -1 78 -3.7164933979511261e-02</internalNodes>
    <leafValues>
      6.0323464870452881e-01 -2.2045797109603882e-01</leafValues></_>
  <_>
    <internalNodes>
      0 -1 98 2.7502819895744324e-02</internalNodes>
    <leafValues>
      -1.8909089267253876e-01 6.8409961462020874e-01</leafValues></_>
  <_>
    <internalNodes>
      0 -1 88 2.4843994528055191e-02</internalNodes>
    <leafValues>
      -2.0602753758430481e-01 7.7499717473983765e-01</leafValues></_>
  <_>
    <internalNodes>
      0 -1 45 -3.4762769937515259e-03</internalNodes>
    <leafValues>
      5.0000065565109253e-01 -3.3975502848625183e-
01</leafValues></_></weakClassifiers></_>
<!-- stage 6 -->
<_>
  <maxWeakCount>6</maxWeakCount>
  <stageThreshold>-6.5365713834762573e-01</stageThreshold>
  <weakClassifiers>
    <_>
      <internalNodes>
        0 -1 18 -5.0435282289981842e-02</internalNodes>
      <leafValues>
        5.0943398475646973e-01 -3.7500000000000000e-01</leafValues></_>
    <_>
      <internalNodes>
        0 -1 37 1.0803389362990856e-02</internalNodes>

```

```

    <leafValues>
      -2.3136703670024872e-01 5.4415243864059448e-01</leafValues></_>
  <_>
    <internalNodes>
      0 -1 72 1.3393448665738106e-02</internalNodes>
    <leafValues>
      -2.1411561965942383e-01 5.9532040357589722e-01</leafValues></_>
  <_>
    <internalNodes>
      0 -1 21 1.2785756960511208e-02</internalNodes>
    <leafValues>
      -2.5409993529319763e-01 5.5940347909927368e-01</leafValues></_>
  <_>
    <internalNodes>
      0 -1 119 5.4867949802428484e-04</internalNodes>
    <leafValues>
      2.3524509370326996e-01 -8.2357507944107056e-01</leafValues></_>
  <_>
    <internalNodes>
      0 -1 91 5.4272037232294679e-05</internalNodes>
    <leafValues>
      -3.4452232718467712e-01 4.9470311403274536e-
01</leafValues></_></weakClassifiers></_>
  <!-- stage 7 -->
  <_>
    <maxWeakCount>9</maxWeakCount>
    <stageThreshold>-6.5736699104309082e-01</stageThreshold>
    <weakClassifiers>
      <_>
        <internalNodes>
          0 -1 3 -1.3526774942874908e-01</internalNodes>
        <leafValues>
          8.2352942228317261e-01 -3.5395190119743347e-01</leafValues></_>
      <_>
        <internalNodes>
          0 -1 67 1.2002856965409592e-04</internalNodes>
        <leafValues>
          -3.3729448914527893e-01 4.0744572877883911e-01</leafValues></_>
      <_>
        <internalNodes>
          0 -1 33 3.1994633376598358e-02</internalNodes>
        <leafValues>
          -1.4769463241100311e-01 6.7252898216247559e-01</leafValues></_>
    </weakClassifiers>
  </_>

```

```

<_>
  <internalNodes>
    0 -1 31 3.5345903597772121e-03</internalNodes>
  <leafValues>
    -2.4748131632804871e-01 4.6998697519302368e-01</leafValues></_>
<_>
  <internalNodes>
    0 -1 119 -6.7092257086187601e-04</internalNodes>
  <leafValues>
    -7.0437061786651611e-01 2.0970605313777924e-01</leafValues></_>
<_>
  <internalNodes>
    0 -1 66 -6.1032120138406754e-03</internalNodes>
  <leafValues>
    5.9277254343032837e-01 -2.4425935745239258e-01</leafValues></_>
<_>
  <internalNodes>
    0 -1 110 -1.1322286445647478e-03</internalNodes>
  <leafValues>
    5.9035724401473999e-01 -2.3792448639869690e-01</leafValues></_>
<_>
  <internalNodes>
    0 -1 55 1.5436533093452454e-01</internalNodes>
  <leafValues>
    -1.7048406600952148e-01 7.8326946496963501e-01</leafValues></_>
<_>
  <internalNodes>
    0 -1 1 -1.0820342868100852e-04</internalNodes>
  <leafValues>
    -5.9019136428833008e-01 2.9158961772918701e-
01</leafValues></_></weakClassifiers></_>
<!-- stage 8 -->
<_>
  <maxWeakCount>9</maxWeakCount>
  <stageThreshold>-8.3958202600479126e-01</stageThreshold>
  <weakClassifiers>
    <_>
      <internalNodes>
        0 -1 8 2.1527592092752457e-02</internalNodes>
      <leafValues>
        -3.2225912809371948e-01 9.1666668653488159e-01</leafValues></_>
    <_>
      <internalNodes>

```

```

    0 -1 115 8.3945989608764648e-03</internalNodes>
    <leafValues>
    1.7609061300754547e-01 -6.6954767704010010e-01</leafValues></_>
  <_>
    <internalNodes>
    0 -1 15 -1.9980578217655420e-03</internalNodes>
    <leafValues>
    -5.0369691848754883e-01 2.5559532642364502e-01</leafValues></_>
  <_>
    <internalNodes>
    0 -1 16 1.0273668915033340e-02</internalNodes>
    <leafValues>
    2.3170879483222961e-01 -6.3879901170730591e-01</leafValues></_>
  <_>
    <internalNodes>
    0 -1 117 -1.6623027622699738e-03</internalNodes>
    <leafValues>
    -7.4079960584640503e-01 1.6420131921768188e-01</leafValues></_>
  <_>
    <internalNodes>
    0 -1 96 3.3945874747587368e-05</internalNodes>
    <leafValues>
    -3.9398622512817383e-01 3.5643091797828674e-01</leafValues></_>
  <_>
    <internalNodes>
    0 -1 69 1.6717996913939714e-05</internalNodes>
    <leafValues>
    -3.4098026156425476e-01 4.3391165137290955e-01</leafValues></_>
  <_>
    <internalNodes>
    0 -1 2 1.6276888549327850e-02</internalNodes>
    <leafValues>
    -2.4513623118400574e-01 6.4181029796600342e-01</leafValues></_>
  <_>
    <internalNodes>
    0 -1 107 -1.8433205783367157e-02</internalNodes>
    <leafValues>
    7.5313591957092285e-01 -2.1023242175579071e-
01</leafValues></_></weakClassifiers></_>
<!-- stage 9 -->
<_>
  <maxWeakCount>11</maxWeakCount>
  <stageThreshold>-9.4963455200195313e-01</stageThreshold>

```

```

<weakClassifiers>
  <_>
    <internalNodes>
      0 -1 89 2.3219041526317596e-02</internalNodes>
    <leafValues>
      -4.2622950673103333e-01 3.5802468657493591e-01</leafValues></_>
  <_>
    <internalNodes>
      0 -1 9 -1.4739182591438293e-01</internalNodes>
    <leafValues>
      -6.7523020505905151e-01 1.6210632026195526e-01</leafValues></_>
  <_>
    <internalNodes>
      0 -1 80 1.4052016194909811e-03</internalNodes>
    <leafValues>
      2.6192378997802734e-01 -4.7646513581275940e-01</leafValues></_>
  <_>
    <internalNodes>
      0 -1 71 -1.3184313429519534e-03</internalNodes>
    <leafValues>
      -6.6755521297454834e-01 2.3899655044078827e-01</leafValues></_>
  <_>
    <internalNodes>
      0 -1 99 1.3012244016863406e-04</internalNodes>
    <leafValues>
      -3.6937472224235535e-01 3.7068396806716919e-01</leafValues></_>
  <_>
    <internalNodes>
      0 -1 23 -3.4093558788299561e-02</internalNodes>
    <leafValues>
      6.3756501674652100e-01 -2.3725315928459167e-01</leafValues></_>
  <_>
    <internalNodes>
      0 -1 75 -1.3889635447412729e-03</internalNodes>
    <leafValues>
      5.6103795766830444e-01 -2.1320627629756927e-01</leafValues></_>
  <_>
    <internalNodes>
      0 -1 112 -2.1385720174293965e-04</internalNodes>
    <leafValues>
      -5.4190242290496826e-01 2.6918718218803406e-01</leafValues></_>
  <_>
    <internalNodes>

```

```

    0 -1 68 -1.8415417522192001e-02</internalNodes>
    <leafValues>
    5.2167558670043945e-01 -2.7002996206283569e-01</leafValues></_>
  <_>
    <internalNodes>
    0 -1 64 -6.0182421293575317e-05</internalNodes>
    <leafValues>
    3.7073376774787903e-01 -4.4004610180854797e-01</leafValues></_>
  <_>
    <internalNodes>
    0 -1 7 5.4631521925330162e-04</internalNodes>
    <leafValues>
    2.4448370933532715e-01 -6.5432184934616089e-
01</leafValues></_></weakClassifiers></_>
  <!-- stage 10 -->
  <_>
    <maxWeakCount>7</maxWeakCount>
    <stageThreshold>-7.4304336309432983e-01</stageThreshold>
    <weakClassifiers>
    <_>
      <internalNodes>
      0 -1 48 -1.7979476600885391e-02</internalNodes>
      <leafValues>
      6.2500000000000000e-01 -3.7906137108802795e-01</leafValues></_>
    <_>
      <internalNodes>
      0 -1 36 1.9815906882286072e-02</internalNodes>
      <leafValues>
      -3.1149134039878845e-01 5.2956390380859375e-01</leafValues></_>
    <_>
      <internalNodes>
      0 -1 87 3.4883656189776957e-04</internalNodes>
      <leafValues>
      2.3883788287639618e-01 -6.8119370937347412e-01</leafValues></_>
    <_>
      <internalNodes>
      0 -1 59 -9.8260911181569099e-04</internalNodes>
      <leafValues>
      5.0640410184860229e-01 -2.8138798475265503e-01</leafValues></_>
    <_>
      <internalNodes>
      0 -1 97 4.7599506797268987e-04</internalNodes>
      <leafValues>

```

```

    3.2087877392768860e-01 -4.9866375327110291e-01</leafValues></_>
<_>
  <internalNodes>
    0 -1 39 -7.0969064836390316e-05</internalNodes>
  <leafValues>
    4.1990715265274048e-01 -3.2152491807937622e-01</leafValues></_>
<_>
  <internalNodes>
    0 -1 47 2.4920729920268059e-03</internalNodes>
  <leafValues>
    2.5714927911758423e-01 -6.3999968767166138e-
01</leafValues></_></weakClassifiers></_>
<!-- stage 11 -->
<_>
  <maxWeakCount>11</maxWeakCount>
  <stageThreshold>-9.1353237628936768e-01</stageThreshold>
  <weakClassifiers>
    <_>
      <internalNodes>
        0 -1 95 1.3367688283324242e-02</internalNodes>
      <leafValues>
        -4.0890687704086304e-01 3.3333334326744080e-01</leafValues></_>
    <_>
      <internalNodes>
        0 -1 5 -1.9295751117169857e-03</internalNodes>
      <leafValues>
        -3.9656910300254822e-01 2.8916656970977783e-01</leafValues></_>
    <_>
      <internalNodes>
        0 -1 101 3.0517242848873138e-03</internalNodes>
      <leafValues>
        1.3902786374092102e-01 -7.9475659132003784e-01</leafValues></_>
    <_>
      <internalNodes>
        0 -1 70 3.1974979210644960e-03</internalNodes>
      <leafValues>
        -2.4648807942867279e-01 4.8057097196578979e-01</leafValues></_>
    <_>
      <internalNodes>
        0 -1 108 -4.3735187500715256e-04</internalNodes>
      <leafValues>
        -5.3437834978103638e-01 2.1633844077587128e-01</leafValues></_>
  <_>

```



```

    <internalNodes>
      0 -1 84 3.1526302918791771e-03</internalNodes>
    <leafValues>
      1.5644282102584839e-01 -6.7118269205093384e-01</leafValues></_>
  <_>
    <internalNodes>
      0 -1 46 2.0084698917344213e-05</internalNodes>
    <leafValues>
      -3.7831410765647888e-01 2.8954190015792847e-01</leafValues></_>
  <_>
    <internalNodes>
      0 -1 4 -2.5178951909765601e-04</internalNodes>
    <leafValues>
      -4.3407770991325378e-01 2.5835114717483521e-01</leafValues></_>
  <_>
    <internalNodes>
      0 -1 35 2.4880815180949867e-04</internalNodes>
    <leafValues>
      -3.1430104374885559e-01 3.7156143784523010e-01</leafValues></_>
  <_>
    <internalNodes>
      0 -1 32 -4.8759959638118744e-02</internalNodes>
    <leafValues>
      8.4794360399246216e-01 -1.4261768758296967e-01</leafValues></_>
  <_>
    <internalNodes>
      0 -1 25 2.4241961000370793e-05</internalNodes>
    <leafValues>
      3.4539404511451721e-01 -3.2169523835182190e-
01</leafValues></_></weakClassifiers></_>
  <!-- stage 12 -->
  <_>
    <maxWeakCount>10</maxWeakCount>
    <stageThreshold>-6.9279474020004272e-01</stageThreshold>
    <weakClassifiers>
      <_>
        <internalNodes>
          0 -1 42 -1.0597408749163151e-02</internalNodes>
        <leafValues>
          4.4444444775581360e-01 -3.6531364917755127e-01</leafValues></_>
      <_>
        <internalNodes>
          0 -1 13 -5.5487409234046936e-02</internalNodes>

```

```

<leafValues>
-8.7996333837509155e-01 1.0403890907764435e-01</leafValues></_>
<_>
<internalNodes>
0 -1 34 1.3629233464598656e-02</internalNodes>
<leafValues>
-1.7889595031738281e-01 6.0624015331268311e-01</leafValues></_>
<_>
<internalNodes>
0 -1 49 -2.4555999785661697e-02</internalNodes>
<leafValues>
4.8026964068412781e-01 -3.1814077496528625e-01</leafValues></_>
<_>
<internalNodes>
0 -1 104 3.8431509165093303e-04</internalNodes>
<leafValues>
-2.6850143074989319e-01 3.8705798983573914e-01</leafValues></_>
<_>
<internalNodes>
0 -1 24 -8.7894842028617859e-02</internalNodes>
<leafValues>
5.9813231229782104e-01 -2.0377379655838013e-01</leafValues></_>
<_>
<internalNodes>
0 -1 113 -1.6546586266485974e-05</internalNodes>
<leafValues>
-4.2872896790504456e-01 2.5290644168853760e-01</leafValues></_>
<_>
<internalNodes>
0 -1 52 1.0849102400243282e-03</internalNodes>
<leafValues>
-3.0418625473976135e-01 4.8468145728111267e-01</leafValues></_>
<_>
<internalNodes>
0 -1 12 -1.8676000181585550e-03</internalNodes>
<leafValues>
4.7259736061096191e-01 -2.2560799121856689e-01</leafValues></_>
<_>
<internalNodes>
0 -1 14 -6.9200986763462424e-04</internalNodes>
<leafValues>
-6.3929003477096558e-01 1.8561150133609772e-
01</leafValues></_></weakClassifiers></_>

```

```

<!-- stage 13 -->
<_>
  <maxWeakCount>7</maxWeakCount>
  <stageThreshold>-5.7735335826873779e-01</stageThreshold>
  <weakClassifiers>
    <_>
      <internalNodes>
        0 -1 60 1.7728971317410469e-02</internalNodes>
      <leafValues>
        -3.2659932971000671e-01 7.8571426868438721e-01</leafValues></_>
    <_>
      <internalNodes>
        0 -1 56 -9.5284003764390945e-03</internalNodes>
      <leafValues>
        5.1888865232467651e-01 -2.6376354694366455e-01</leafValues></_>
    <_>
      <internalNodes>
        0 -1 11 -2.0160281565040350e-03</internalNodes>
      <leafValues>
        -5.8295178413391113e-01 2.5967410206794739e-01</leafValues></_>
    <_>
      <internalNodes>
        0 -1 10 1.5735668421257287e-04</internalNodes>
      <leafValues>
        2.7167066931724548e-01 -4.4707870483398438e-01</leafValues></_>
    <_>
      <internalNodes>
        0 -1 20 -4.4289809011388570e-05</internalNodes>
      <leafValues>
        -4.2379966378211975e-01 3.1767934560775757e-01</leafValues></_>
    <_>
      <internalNodes>
        0 -1 111 8.6165280663408339e-05</internalNodes>
      <leafValues>
        2.2165802121162415e-01 -4.8844155669212341e-01</leafValues></_>
    <_>
      <internalNodes>
        0 -1 40 9.8472344689071178e-04</internalNodes>
      <leafValues>
        -2.1504674851894379e-01 5.7308125495910645e-
01</leafValues></_></weakClassifiers></_>
  <!-- stage 14 -->
  <_>

```

```

<maxWeakCount>10</maxWeakCount>
<stageThreshold>-1.0302747488021851e+00</stageThreshold>
<weakClassifiers>
  <_>
    <internalNodes>
      0 -1 62 -7.1634396910667419e-02</internalNodes>
    <leafValues>
      6.3636362552642822e-01 -4.0740740299224854e-01</leafValues></_>
  <_>
    <internalNodes>
      0 -1 109 -9.1491913190111518e-04</internalNodes>
    <leafValues>
      -7.7948713302612305e-01 1.5945658087730408e-01</leafValues></_>
  <_>
    <internalNodes>
      0 -1 105 -2.8533106669783592e-02</internalNodes>
    <leafValues>
      7.2286880016326904e-01 -1.8612043559551239e-01</leafValues></_>
  <_>
    <internalNodes>
      0 -1 0 -6.1496254056692123e-04</internalNodes>
    <leafValues>
      -7.9236334562301636e-01 1.7380641400814056e-01</leafValues></_>
  <_>
    <internalNodes>
      0 -1 61 3.5112433135509491e-02</internalNodes>
    <leafValues>
      -2.5222244858741760e-01 4.6221646666526794e-01</leafValues></_>
  <_>
    <internalNodes>
      0 -1 77 -1.5692582819610834e-03</internalNodes>
    <leafValues>
      4.8103633522987366e-01 -2.5807148218154907e-01</leafValues></_>
  <_>
    <internalNodes>
      0 -1 116 6.3937430968508124e-04</internalNodes>
    <leafValues>
      1.9300635159015656e-01 -6.6959172487258911e-01</leafValues></_>
  <_>
    <internalNodes>
      0 -1 54 1.4973044744692743e-04</internalNodes>
    <leafValues>
      -4.0527614951133728e-01 3.5537254810333252e-01</leafValues></_>

```

```

    <_>
    <internalNodes>
    0 -1 27 4.3619658797979355e-02</internalNodes>
    <leafValues>
    -1.6291403770446777e-01 8.4390562772750854e-01</leafValues></_>
    <_>
    <internalNodes>
    0 -1 100 9.4782830274198204e-05</internalNodes>
    <leafValues>
    2.6648682355880737e-01 -4.9702170491218567e-
01</leafValues></_></weakClassifiers></_></stages>
<features>
    <_>
    <rects>
    <_>
    0 0 2 1 -1.</_>
    <_>
    1 0 1 1 2.</_></rects>
    <tilted>0</tilted></_>
    <_>
    <rects>
    <_>
    0 1 1 9 -1.</_>
    <_>
    0 4 1 3 3.</_></rects>
    <tilted>0</tilted></_>
    <_>
    <rects>
    <_>
    0 2 12 3 -1.</_>
    <_>
    4 2 4 3 3.</_></rects>
    <tilted>0</tilted></_>
    <_>
    <rects>
    <_>
    0 3 8 19 -1.</_>
    <_>
    4 3 4 19 2.</_></rects>
    <tilted>0</tilted></_>
    <_>
    <rects>
    <_>

```

```

    0 4 4 2 -1.</_>
  <_>
    0 4 2 1 2.</_>
  <_>
    2 5 2 1 2.</_></rects>
  <tilted>0</tilted></_>
<_>
  <rects>
    <_>
      0 4 2 16 -1.</_>
    <_>
      0 12 2 8 2.</_></rects>
    <tilted>0</tilted></_>
  <_>
    <rects>
      <_>
        0 6 12 9 -1.</_>
      <_>
        0 9 12 3 3.</_></rects>
      <tilted>0</tilted></_>
    <_>
      <rects>
        <_>
          0 11 2 3 -1.</_>
        <_>
          0 12 2 1 3.</_></rects>
        <tilted>0</tilted></_>
      <_>
        <rects>
          <_>
            0 13 4 7 -1.</_>
          <_>
            2 13 2 7 2.</_></rects>
          <tilted>0</tilted></_>
        <_>
          <rects>
            <_>
              0 13 24 11 -1.</_>
            <_>
              12 13 12 11 2.</_></rects>
            <tilted>0</tilted></_>
          <_>
            <rects>

```

```

    <_>
    0 14 1 6 -1.</_>
    <_>
    0 17 1 3 2.</_></rects>
    <tilted>0</tilted></_>
<_>
<rects>
    <_>
    0 14 1 9 -1.</_>
    <_>
    0 17 1 3 3.</_></rects>
    <tilted>0</tilted></_>
<_>
<rects>
    <_>
    0 20 16 2 -1.</_>
    <_>
    0 20 8 1 2.</_>
    <_>
    8 21 8 1 2.</_></rects>
    <tilted>0</tilted></_>
<_>
<rects>
    <_>
    0 21 22 3 -1.</_>
    <_>
    11 21 11 3 2.</_></rects>
    <tilted>0</tilted></_>
<_>
<rects>
    <_>
    1 0 4 1 -1.</_>
    <_>
    3 0 2 1 2.</_></rects>
    <tilted>0</tilted></_>
<_>
<rects>
    <_>
    1 0 16 1 -1.</_>
    <_>
    9 0 8 1 2.</_></rects>
    <tilted>0</tilted></_>
<_>

```

```

<rects>
  <_>
    1 0 22 1 -1.</_>
  <_>
    12 0 11 1 2.</_></rects>
  <tilted>0</tilted></_>
<_>
  <rects>
    <_>
      1 8 6 14 -1.</_>
    <_>
      1 8 3 7 2.</_>
    <_>
      4 15 3 7 2.</_></rects>
    <tilted>0</tilted></_>
<_>
  <rects>
    <_>
      1 8 15 7 -1.</_>
    <_>
      6 8 5 7 3.</_></rects>
    <tilted>0</tilted></_>
<_>
  <rects>
    <_>
      1 11 6 5 -1.</_>
    <_>
      3 11 2 5 3.</_></rects>
    <tilted>0</tilted></_>
<_>
  <rects>
    <_>
      1 13 1 8 -1.</_>
    <_>
      1 17 1 4 2.</_></rects>
    <tilted>0</tilted></_>
<_>
  <rects>
    <_>
      1 13 12 2 -1.</_>
    <_>
      5 13 4 2 3.</_></rects>
    <tilted>0</tilted></_>

```



```

<_>
  <rects>
    <_>
      1 14 9 4 -1.</_>
    <_>
      4 14 3 4 3.</_></rects>
    <tilted>0</tilted></_>
  <_>
    <rects>
      <_>
        1 14 16 10 -1.</_>
      <_>
        1 14 8 5 2.</_>
      <_>
        9 19 8 5 2.</_></rects>
      <tilted>0</tilted></_>
    <_>
      <rects>
        <_>
          1 14 18 9 -1.</_>
        <_>
          10 14 9 9 2.</_></rects>
        <tilted>0</tilted></_>
      <_>
        <rects>
          <_>
            1 20 3 2 -1.</_>
          <_>
            2 20 1 2 3.</_></rects>
          <tilted>0</tilted></_>
        <_>
          <rects>
            <_>
              2 3 2 8 -1.</_>
            <_>
              2 3 1 4 2.</_>
            <_>
              3 7 1 4 2.</_></rects>
            <tilted>0</tilted></_>
          <_>
            <rects>
              <_>
                2 6 10 14 -1.</_>

```

```

    <_>
    2 6 5 7 2.</_>
    <_>
    7 13 5 7 2.</_></rects>
    <tilted>0</tilted></_>
<_>
<rects>
    <_>
    2 8 4 15 -1.</_>
    <_>
    4 8 2 15 2.</_></rects>
    <tilted>0</tilted></_>
<_>
<rects>
    <_>
    2 10 9 5 -1.</_>
    <_>
    5 10 3 5 3.</_></rects>
    <tilted>0</tilted></_>
<_>
<rects>
    <_>
    2 11 9 2 -1.</_>
    <_>
    5 11 3 2 3.</_></rects>
    <tilted>0</tilted></_>
<_>
<rects>
    <_>
    2 20 8 4 -1.</_>
    <_>
    2 22 8 2 2.</_></rects>
    <tilted>0</tilted></_>
<_>
<rects>
    <_>
    3 1 8 18 -1.</_>
    <_>
    3 1 4 9 2.</_>
    <_>
    7 10 4 9 2.</_></rects>
    <tilted>0</tilted></_>
<_>

```

```

<rects>
  <_>
    3 17 15 6 -1.</_>
  <_>
    3 19 15 2 3.</_></rects>
<tilted>0</tilted></_>
<_>
  <rects>
    <_>
      4 0 15 4 -1.</_>
    <_>
      4 2 15 2 2.</_></rects>
  <tilted>0</tilted></_>
<_>
  <rects>
    <_>
      5 0 2 6 -1.</_>
    <_>
      5 2 2 2 3.</_></rects>
  <tilted>0</tilted></_>
<_>
  <rects>
    <_>
      5 0 12 8 -1.</_>
    <_>
      5 4 12 4 2.</_></rects>
  <tilted>0</tilted></_>
<_>
  <rects>
    <_>
      5 1 14 4 -1.</_>
    <_>
      5 3 14 2 2.</_></rects>
  <tilted>0</tilted></_>
<_>
  <rects>
    <_>
      5 3 6 9 -1.</_>
    <_>
      7 3 2 9 3.</_></rects>
  <tilted>0</tilted></_>
<_>
  <rects>

```

```

    <_>
    5 10 1 2 -1.</_>
    <_>
    5 11 1 1 2.</_></rects>
    <tilted>0</tilted></_>
<_>
<rects>
    <_>
    5 12 3 1 -1.</_>
    <_>
    6 12 1 1 3.</_></rects>
    <tilted>0</tilted></_>
<_>
<rects>
    <_>
    5 15 2 2 -1.</_>
    <_>
    6 15 1 2 2.</_></rects>
    <tilted>0</tilted></_>
<_>
<rects>
    <_>
    5 21 17 3 -1.</_>
    <_>
    5 22 17 1 3.</_></rects>
    <tilted>0</tilted></_>
<_>
<rects>
    <_>
    6 4 18 15 -1.</_>
    <_>
    6 9 18 5 3.</_></rects>
    <tilted>0</tilted></_>
<_>
<rects>
    <_>
    6 15 3 1 -1.</_>
    <_>
    7 15 1 1 3.</_></rects>
    <tilted>0</tilted></_>
<_>
<rects>
    <_>

```

```

        6 15 4 8 -1.</_>
    <_>
        6 19 4 4 2.</_></rects>
    <tilted>0</tilted></_>
<_>
    <rects>
        <_>
            6 18 4 3 -1.</_>
        <_>
            6 19 4 1 3.</_></rects>
        <tilted>0</tilted></_>
    <_>
    <rects>
        <_>
            6 22 9 1 -1.</_>
        <_>
            9 22 3 1 3.</_></rects>
        <tilted>0</tilted></_>
    <_>
    <rects>
        <_>
            7 0 10 4 -1.</_>
        <_>
            7 2 10 2 2.</_></rects>
        <tilted>0</tilted></_>
    <_>
    <rects>
        <_>
            7 0 15 6 -1.</_>
        <_>
            7 3 15 3 2.</_></rects>
        <tilted>0</tilted></_>
    <_>
    <rects>
        <_>
            7 7 6 14 -1.</_>
        <_>
            9 7 2 14 3.</_></rects>
        <tilted>0</tilted></_>
    <_>
    <rects>
        <_>
            7 14 2 2 -1.</_>

```

```

    <_>
    7 14 1 1 2.</_>
    <_>
    8 15 1 1 2.</_></rects>
    <tilted>0</tilted></_>
<_>
<rects>
    <_>
    7 15 6 2 -1.</_>
    <_>
    7 15 3 1 2.</_>
    <_>
    10 16 3 1 2.</_></rects>
    <tilted>0</tilted></_>
<_>
<rects>
    <_>
    7 19 2 2 -1.</_>
    <_>
    7 19 1 1 2.</_>
    <_>
    8 20 1 1 2.</_></rects>
    <tilted>0</tilted></_>
<_>
<rects>
    <_>
    8 0 3 10 -1.</_>
    <_>
    9 0 1 10 3.</_></rects>
    <tilted>0</tilted></_>
<_>
<rects>
    <_>
    8 0 12 24 -1.</_>
    <_>
    8 8 12 8 3.</_></rects>
    <tilted>0</tilted></_>
<_>
<rects>
    <_>
    8 1 12 6 -1.</_>
    <_>
    8 3 12 2 3.</_></rects>

```

```

    <tilted>0</tilted></_>
  <_>
    <rects>
      <_>
        8 3 4 8 -1.</_>
      <_>
        10 3 2 8 2.</_></rects>
    <tilted>0</tilted></_>
  <_>
    <rects>
      <_>
        8 5 2 4 -1.</_>
      <_>
        8 5 1 2 2.</_>
      <_>
        9 7 1 2 2.</_></rects>
    <tilted>0</tilted></_>
  <_>
    <rects>
      <_>
        8 5 2 6 -1.</_>
      <_>
        8 5 1 3 2.</_>
      <_>
        9 8 1 3 2.</_></rects>
    <tilted>0</tilted></_>
  <_>
    <rects>
      <_>
        8 22 13 2 -1.</_>
      <_>
        8 23 13 1 2.</_></rects>
    <tilted>0</tilted></_>
  <_>
    <rects>
      <_>
        9 1 9 15 -1.</_>
      <_>
        9 6 9 5 3.</_></rects>
    <tilted>0</tilted></_>
  <_>
    <rects>
      <_>

```

```

    9 11 15 8 -1.</_>
  <_>
    14 11 5 8 3.</_></rects>
  <tilted>0</tilted></_>
<_>
  <rects>
    <_>
      9 15 15 2 -1.</_>
    <_>
      14 15 5 2 3.</_></rects>
    <tilted>0</tilted></_>
  <_>
    <rects>
      <_>
        9 17 10 2 -1.</_>
      <_>
        9 18 10 1 2.</_></rects>
      <tilted>0</tilted></_>
    <_>
      <rects>
        <_>
          9 19 4 4 -1.</_>
        <_>
          9 19 2 2 2.</_>
        <_>
          11 21 2 2 2.</_></rects>
        <tilted>0</tilted></_>
      <_>
        <rects>
          <_>
            10 0 8 6 -1.</_>
          <_>
            10 0 4 3 2.</_>
          <_>
            14 3 4 3 2.</_></rects>
          <tilted>0</tilted></_>
        <_>
          <rects>
            <_>
              10 2 2 3 -1.</_>
            <_>
              10 3 2 1 3.</_></rects>
          <tilted>0</tilted></_>

```



```

<_>
  <rects>
    <_>
      10 13 14 10 -1.</_>
    <_>
      10 13 7 5 2.</_>
    <_>
      17 18 7 5 2.</_></rects>
  <tilted>0</tilted></_>
<_>
  <rects>
    <_>
      10 16 2 4 -1.</_>
    <_>
      10 16 1 2 2.</_>
    <_>
      11 18 1 2 2.</_></rects>
  <tilted>0</tilted></_>
<_>
  <rects>
    <_>
      10 19 9 3 -1.</_>
    <_>
      10 20 9 1 3.</_></rects>
  <tilted>0</tilted></_>
<_>
  <rects>
    <_>
      11 0 6 1 -1.</_>
    <_>
      14 0 3 1 2.</_></rects>
  <tilted>0</tilted></_>
<_>
  <rects>
    <_>
      11 0 12 6 -1.</_>
    <_>
      11 0 6 3 2.</_>
    <_>
      17 3 6 3 2.</_></rects>
  <tilted>0</tilted></_>
<_>
  <rects>

```

```

<_>
  11 1 4 8 -1.</_>
<_>
  11 1 2 4 2.</_>
<_>
  13 5 2 4 2.</_></rects>
<tilted>0</tilted></_>
<_>
<rects>
  <_>
    11 2 9 4 -1.</_>
  <_>
    14 2 3 4 3.</_></rects>
  <tilted>0</tilted></_>
<_>
<rects>
  <_>
    11 4 2 10 -1.</_>
  <_>
    11 4 1 5 2.</_>
  <_>
    12 9 1 5 2.</_></rects>
  <tilted>0</tilted></_>
<_>
<rects>
  <_>
    11 4 6 2 -1.</_>
  <_>
    13 4 2 2 3.</_></rects>
  <tilted>0</tilted></_>
<_>
<rects>
  <_>
    11 5 3 12 -1.</_>
  <_>
    11 9 3 4 3.</_></rects>
  <tilted>0</tilted></_>
<_>
<rects>
  <_>
    11 12 11 12 -1.</_>
  <_>
    11 16 11 4 3.</_></rects>

```

```

    <tilted>0</tilted></_>
  <_>
    <rects>
      <_>
        11 17 1 6 -1.</_>
      <_>
        11 19 1 2 3.</_></rects>
    <tilted>0</tilted></_>
  <_>
    <rects>
      <_>
        12 0 9 1 -1.</_>
      <_>
        15 0 3 1 3.</_></rects>
    <tilted>0</tilted></_>
  <_>
    <rects>
      <_>
        12 0 12 1 -1.</_>
      <_>
        16 0 4 1 3.</_></rects>
    <tilted>0</tilted></_>
  <_>
    <rects>
      <_>
        13 3 3 16 -1.</_>
      <_>
        14 3 1 16 3.</_></rects>
    <tilted>0</tilted></_>
  <_>
    <rects>
      <_>
        13 5 2 6 -1.</_>
      <_>
        13 5 1 3 2.</_>
      <_>
        14 8 1 3 2.</_></rects>
    <tilted>0</tilted></_>
  <_>
    <rects>
      <_>
        13 6 1 8 -1.</_>
      <_>

```

```

    13 10 1 4 2.</_></rects>
    <tilted>0</tilted></_>
<_>
    <rects>
        <_>
            13 10 3 2 -1.</_>
        <_>
            14 10 1 2 3.</_></rects>
        <tilted>0</tilted></_>
<_>
    <rects>
        <_>
            13 10 6 10 -1.</_>
        <_>
            13 10 3 5 2.</_>
        <_>
            16 15 3 5 2.</_></rects>
        <tilted>0</tilted></_>
<_>
    <rects>
        <_>
            13 23 2 1 -1.</_>
        <_>
            14 23 1 1 2.</_></rects>
        <tilted>0</tilted></_>
<_>
    <rects>
        <_>
            14 3 6 15 -1.</_>
        <_>
            16 3 2 15 3.</_></rects>
        <tilted>0</tilted></_>
<_>
    <rects>
        <_>
            14 4 10 14 -1.</_>
        <_>
            14 4 5 7 2.</_>
        <_>
            19 11 5 7 2.</_></rects>
        <tilted>0</tilted></_>
<_>
    <rects>

```

```

    <_>
    14 4 10 2 -1.</_>
    <_>
    14 5 10 1 2.</_></rects>
    <tilted>0</tilted></_>
<_>
<rects>
    <_>
    14 6 3 1 -1.</_>
    <_>
    15 6 1 1 3.</_></rects>
    <tilted>0</tilted></_>
<_>
<rects>
    <_>
    14 6 8 6 -1.</_>
    <_>
    14 6 4 3 2.</_>
    <_>
    18 9 4 3 2.</_></rects>
    <tilted>0</tilted></_>
<_>
<rects>
    <_>
    14 10 10 14 -1.</_>
    <_>
    19 10 5 14 2.</_></rects>
    <tilted>0</tilted></_>
<_>
<rects>
    <_>
    14 13 6 5 -1.</_>
    <_>
    16 13 2 5 3.</_></rects>
    <tilted>0</tilted></_>
<_>
<rects>
    <_>
    14 15 10 2 -1.</_>
    <_>
    19 15 5 2 2.</_></rects>
    <tilted>0</tilted></_>
<_>

```

```

<rects>
  <_>
    14 19 6 2 -1.</_>
  <_>
    14 19 3 1 2.</_>
  <_>
    17 20 3 1 2.</_></rects>
<tilted>0</tilted></_>
<_>
<rects>
  <_>
    15 0 6 1 -1.</_>
  <_>
    18 0 3 1 2.</_></rects>
<tilted>0</tilted></_>
<_>
<rects>
  <_>
    15 13 9 9 -1.</_>
  <_>
    18 13 3 9 3.</_></rects>
<tilted>0</tilted></_>
<_>
<rects>
  <_>
    16 9 2 6 -1.</_>
  <_>
    16 11 2 2 3.</_></rects>
<tilted>0</tilted></_>
<_>
<rects>
  <_>
    17 0 4 1 -1.</_>
  <_>
    19 0 2 1 2.</_></rects>
<tilted>0</tilted></_>
<_>
<rects>
  <_>
    17 1 3 3 -1.</_>
  <_>
    18 1 1 3 3.</_></rects>
<tilted>0</tilted></_>

```

```

<_>
  <rects>
    <_>
      17 1 6 8 -1.</_>
    <_>
      17 1 3 4 2.</_>
    <_>
      20 5 3 4 2.</_></rects>
  <tilted>0</tilted></_>
<_>
  <rects>
    <_>
      17 12 4 5 -1.</_>
    <_>
      19 12 2 5 2.</_></rects>
  <tilted>0</tilted></_>
<_>
  <rects>
    <_>
      18 0 3 9 -1.</_>
    <_>
      18 3 3 3 3.</_></rects>
  <tilted>0</tilted></_>
<_>
  <rects>
    <_>
      18 0 3 14 -1.</_>
    <_>
      18 7 3 7 2.</_></rects>
  <tilted>0</tilted></_>
<_>
  <rects>
    <_>
      18 9 6 7 -1.</_>
    <_>
      20 9 2 7 3.</_></rects>
  <tilted>0</tilted></_>
<_>
  <rects>
    <_>
      18 10 6 9 -1.</_>
    <_>
      20 10 2 9 3.</_></rects>

```

```

    <tilted>0</tilted></_>
  <_>
    <rects>
      <_>
        18 23 3 1 -1.</_>
      <_>
        19 23 1 1 3.</_></rects>
    <tilted>0</tilted></_>
  <_>
    <rects>
      <_>
        19 23 2 1 -1.</_>
      <_>
        20 23 1 1 2.</_></rects>
    <tilted>0</tilted></_>
  <_>
    <rects>
      <_>
        20 3 2 8 -1.</_>
      <_>
        20 3 1 4 2.</_>
      <_>
        21 7 1 4 2.</_></rects>
    <tilted>0</tilted></_>
  <_>
    <rects>
      <_>
        21 0 2 1 -1.</_>
      <_>
        22 0 1 1 2.</_></rects>
    <tilted>0</tilted></_>
  <_>
    <rects>
      <_>
        21 3 3 3 -1.</_>
      <_>
        21 4 3 1 3.</_></rects>
    <tilted>0</tilted></_>
  <_>
    <rects>
      <_>
        21 23 3 1 -1.</_>
      <_>

```



```

    22 23 1 1 3.</_></rects>
    <tilted>0</tilted></_>
<_>
    <rects>
        <_>
            22 1 2 3 -1.</_>
        <_>
            22 2 2 1 3.</_></rects>
        <tilted>0</tilted></_>
<_>
    <rects>
        <_>
            22 4 2 15 -1.</_>
        <_>
            22 9 2 5 3.</_></rects>
        <tilted>0</tilted></_>
<_>
    <rects>
        <_>
            22 9 2 2 -1.</_>
        <_>
            22 10 2 1 2.</_></rects>
        <tilted>0</tilted></_>
<_>
    <rects>
        <_>
            23 0 1 6 -1.</_>
        <_>
            23 2 1 2 3.</_></rects>
        <tilted>0</tilted></_>
<_>
    <rects>
        <_>
            23 1 1 22 -1.</_>
        <_>
            23 12 1 11 2.</_></rects>
        <tilted>0</tilted></_>
<_>
    <rects>
        <_>
            23 5 1 3 -1.</_>
        <_>
            23 6 1 1 3.</_></rects>

```

```
<tilted>0</tilted></_>
<_>
<rects>
  <_>
    23 8 1 3 -1.</_>
  <_>
    23 9 1 1 3.</_></rects>
  <tilted>0</tilted></_></features></cascade>
</opencv_storage>
```

#### **4.4.2 Litterdetection.py**

```
import cv2

capture = cv2.VideoCapture(1)

cascade = cv2.CascadeClassifier('garbage.xml')

while(True):
    #Capture frame by frame
    sucess,frame = capture.read()

    #Our operation on the frame come here
    # if frame == empty:
    #else:
    blue = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
    waste = cascade.detectMultiScale(blue, 1.3, 5)

    for (x,y,w,h) in waste:
        cv2.rectangle(frame,(x,y),(x+w,y+h),(255,0,0),2)

    cv2.imshow("video", frame)
    if cv2.waitKey(1) & 0xFF == ord('q'):
        break
# when everything done,release the capture
capture.release()
cv2.destroyAllWindows()
```

# CHAPTER 5

## TESTING

### 5.1 SOFTWARE TESTING

Software testing is a critical element of software quality assurance and represents the ultimate review of specification, design, and code generation.

#### 5.1.1 TESTING OBJECTIVES

- To ensure that during operation the system will perform as per specification.
- TO make sure that system meets the user requirements during operation
- To make sure that during the operation, incorrect input, processing and output will be detected
- To see that when correct inputs are fed to the system the outputs are correct
- To verify that the controls incorporated in the same system as intended
- Testing is a process of executing a program with the intent of finding an error
- A good test case is one that has a high probability of finding an as yet undiscovered error.

The software developed has been tested successfully using the following testing strategies and any errors that are encountered are corrected and again the part of the program or the procedure or function is put to testing until all the errors are removed. A successful test is one that uncovers an as yet undiscovered error.

Note that the result of the system testing will prove that the system is working correctly. It will give confidence to system designer, users of the system, prevent frustration during implementation process etc.

## **5.2 TESTING DESIGN:**

### **5.2.1 White box testing**

White box testing is a testing case design method that uses the control structure of the procedure design to derive test cases. All independent paths in a module are exercised at least once, all logical decisions are exercised at once, execute all loops at boundaries and within their operational bounds exercise internal data structure to ensure their validity. Here the customer is given three chances to enter a valid choice out of the given menu. After which the control exits the current menu.

### **5.2.2 Black Box Testing**

Black Box Testing attempts to find errors in following areas or categories, incorrect or missing functions, interface error, errors in data structures, performance error and initialization and termination error. Here all the input data must match the data type to become a valid entry.

The following are the different tests at various levels:

### **5.2.3 Unit Testing:**

Unit testing is essentially for the verification of the code produced during the coding phase and the goal is to test the internal logic of the module/program. In the Generic code project, the unit testing is done during the coding phase of data entry forms whether the functions are working properly or not. In this phase all the drivers are tested they are rightly connected or not.

### **5.2.4 Integration Testing:**

All the tested modules are combined into sub systems, which are then tested. The goal is to see if the modules are properly integrated, and the emphasis is on the testing interfaces between the modules. In the generic code integration testing is done mainly on table creation module and insertion module.

### **5.2.5 Validation Testing**

This testing concentrates on confirming that the software is error-free in all respects. All the specified validations are verified and the software is subjected to

hard-core testing. It also aims at determining the degree of deviation that exists in the software designed from the specification; they are listed out and are corrected.

### 5.2.6 System Testing

This testing is a series of different tests whose primary is to fully exercise the computer-based system. This involves:

- Implementing the system in a simulated production environment and testing it.
- Introducing errors and testing for error handling.

### 5.2.7 TEST CASES:

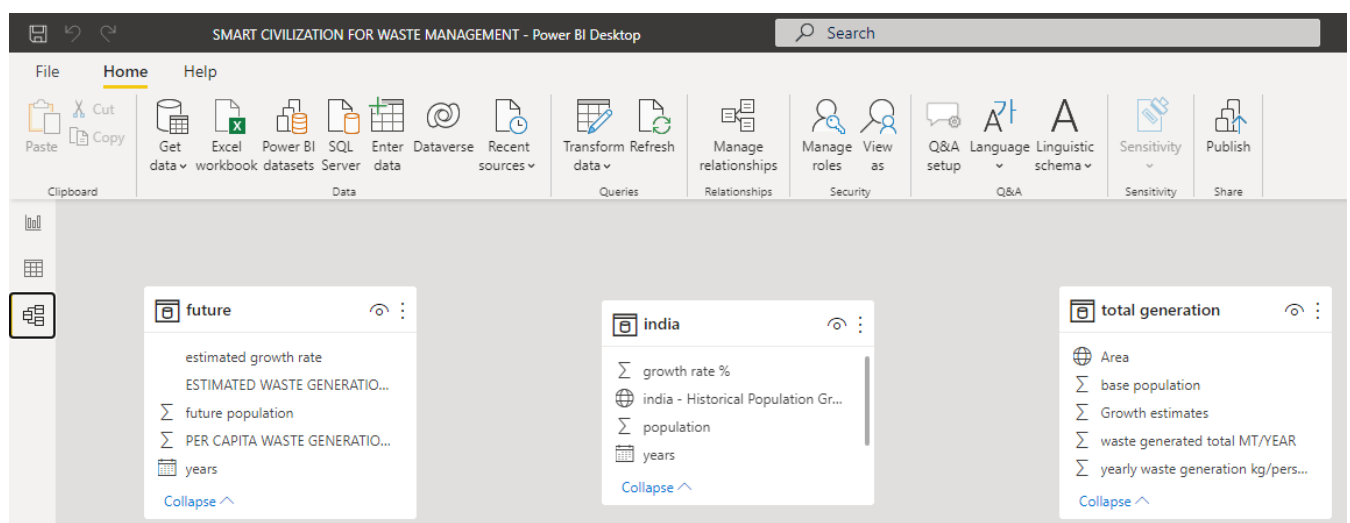
Test case	valid	Invalid
Garbage image detection	Yes	No
Normal image detection	No	Yes

## 5.3 POWER BI

### 5.3.1 PowerBI Model Dashboard

#### PowerBI Model Dashboard

Power BI is a Data Visualization and Business Intelligence tool that converts data from different data sources to interactive dashboards and BI reports. Power BI suite provides multiple software, connector, and services - Power BI desktop, Power BI service based on SaaS, and mobile Power BI apps available for different platforms. These set of services are used by business users to consume data and build BI reports. This tutorial covers all the important concepts in Power BI and provides a foundational understanding on how to use Power BI.



**Fig 5 PowerBI Model Dashboard**

### 5.3.2 Uses of PowerBI:

- A quick start.
- Streamlined publication and distribution.
- Real-time information.
- Ability to customize Power BI app navigation.
- Ability to customize security features.
- Cortana integration.
- Artificial Intelligence.

### 5.3.3 PowerBI Dashboard:

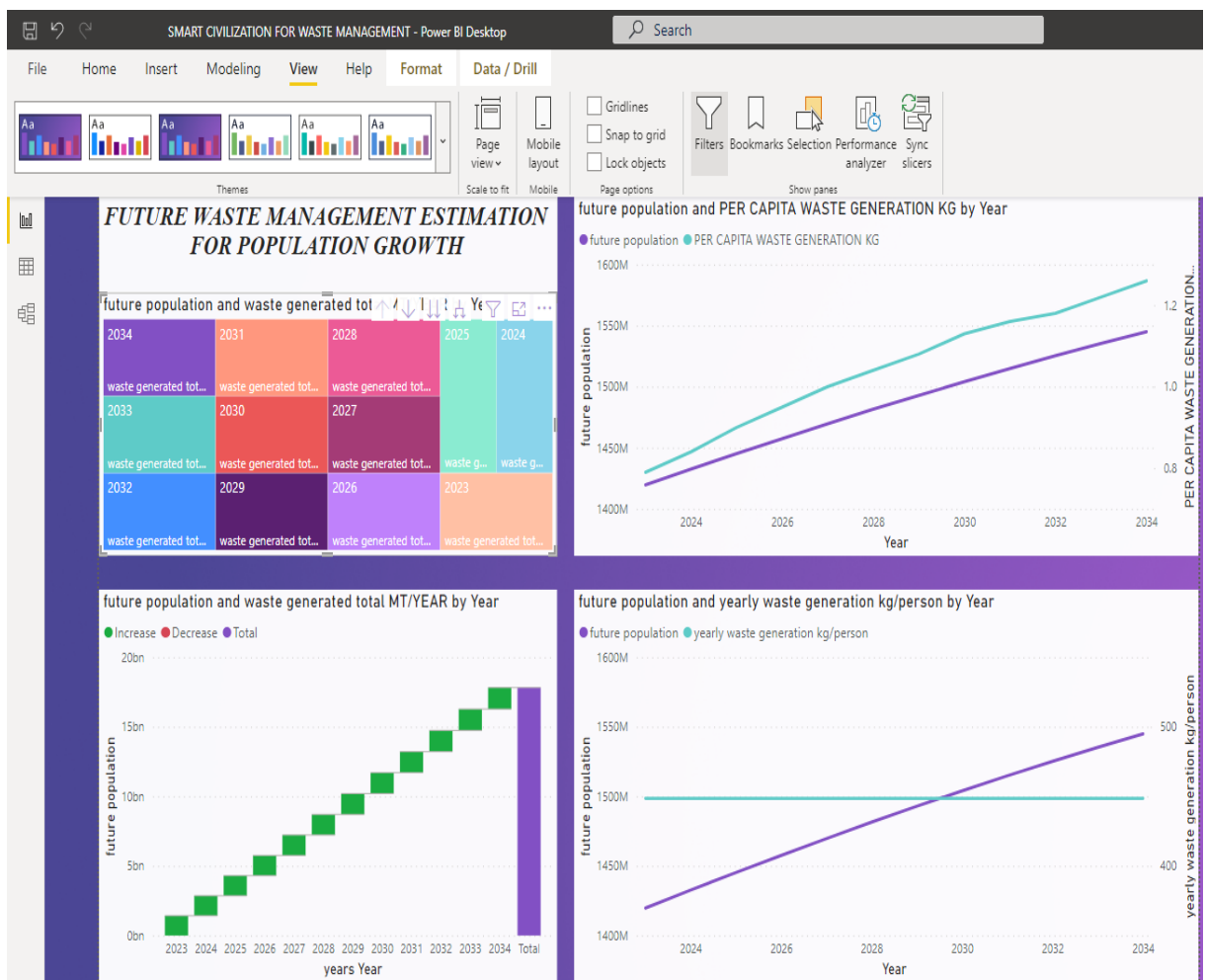
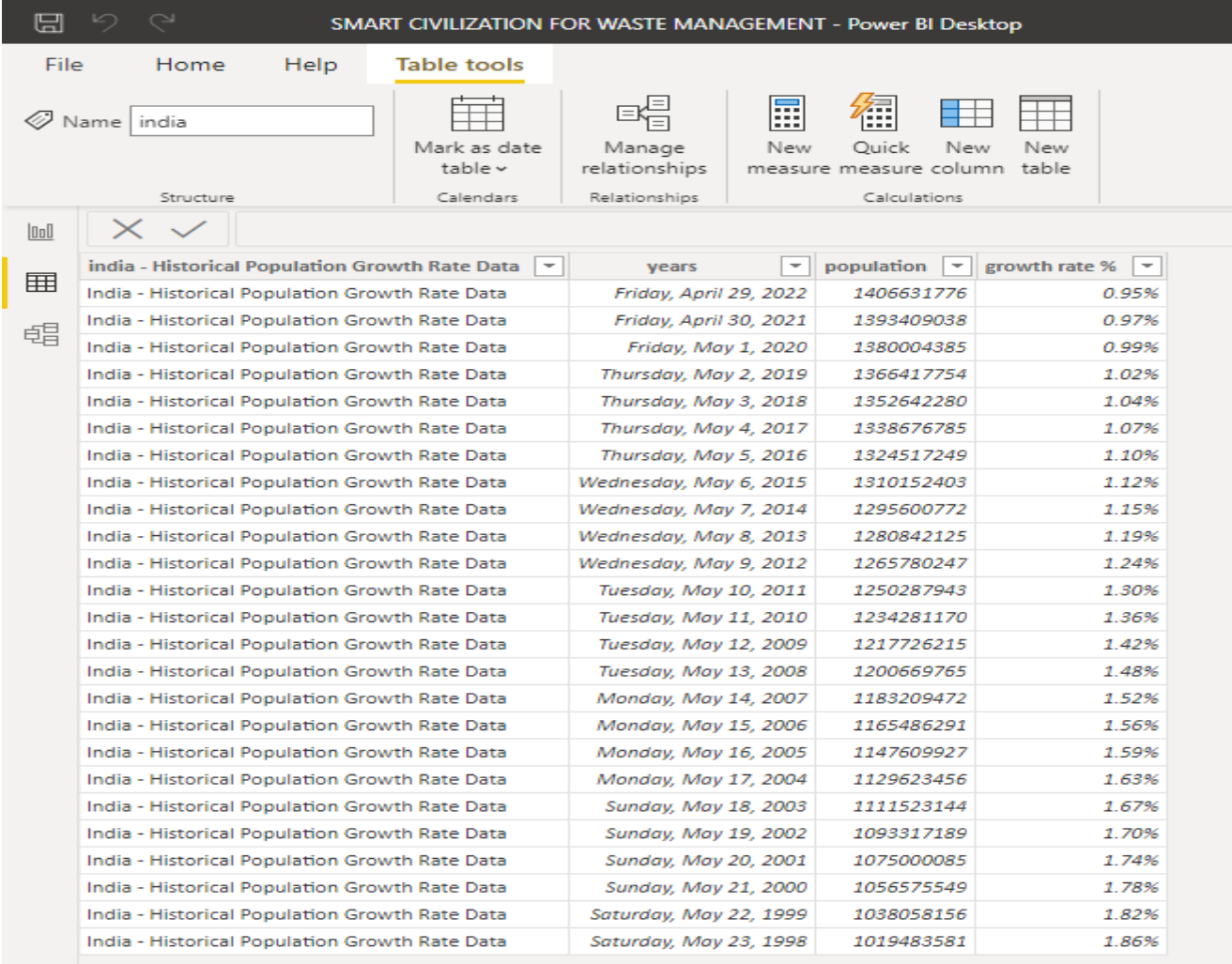


Fig 5.3.3: PowerBI Dashboard.

## CHAPTER 6

### LIST OF IMAGES



The screenshot shows the Power BI Desktop interface with the 'Table tools' ribbon selected. The 'Name' field is set to 'india'. The table displayed is 'india - Historical Population Growth Rate Data' with columns: 'years', 'population', and 'growth rate %'. The table contains 25 rows of data, showing a steady increase in both population and growth rate over time.

india - Historical Population Growth Rate Data	years	population	growth rate %
India - Historical Population Growth Rate Data	Friday, April 29, 2022	1406631776	0.95%
India - Historical Population Growth Rate Data	Friday, April 30, 2021	1393409038	0.97%
India - Historical Population Growth Rate Data	Friday, May 1, 2020	1380004385	0.99%
India - Historical Population Growth Rate Data	Thursday, May 2, 2019	1366417754	1.02%
India - Historical Population Growth Rate Data	Thursday, May 3, 2018	1352642280	1.04%
India - Historical Population Growth Rate Data	Thursday, May 4, 2017	1338676785	1.07%
India - Historical Population Growth Rate Data	Thursday, May 5, 2016	1324517249	1.10%
India - Historical Population Growth Rate Data	Wednesday, May 6, 2015	1310152403	1.12%
India - Historical Population Growth Rate Data	Wednesday, May 7, 2014	1295600772	1.15%
India - Historical Population Growth Rate Data	Wednesday, May 8, 2013	1280842125	1.19%
India - Historical Population Growth Rate Data	Wednesday, May 9, 2012	1265780247	1.24%
India - Historical Population Growth Rate Data	Tuesday, May 10, 2011	1250287943	1.30%
India - Historical Population Growth Rate Data	Tuesday, May 11, 2010	1234281170	1.36%
India - Historical Population Growth Rate Data	Tuesday, May 12, 2009	1217726215	1.42%
India - Historical Population Growth Rate Data	Tuesday, May 13, 2008	1200669765	1.48%
India - Historical Population Growth Rate Data	Monday, May 14, 2007	1183209472	1.52%
India - Historical Population Growth Rate Data	Monday, May 15, 2006	1165486291	1.56%
India - Historical Population Growth Rate Data	Monday, May 16, 2005	1147609927	1.59%
India - Historical Population Growth Rate Data	Monday, May 17, 2004	1129623456	1.63%
India - Historical Population Growth Rate Data	Sunday, May 18, 2003	1111523144	1.67%
India - Historical Population Growth Rate Data	Sunday, May 19, 2002	1093317189	1.70%
India - Historical Population Growth Rate Data	Sunday, May 20, 2001	1075000085	1.74%
India - Historical Population Growth Rate Data	Sunday, May 21, 2000	1056575549	1.78%
India - Historical Population Growth Rate Data	Saturday, May 22, 1999	1038058156	1.82%
India - Historical Population Growth Rate Data	Saturday, May 23, 1998	1019483581	1.86%

**Fig 6.a: India Table.png**

#### 6.1 India Table.png

The population of India has grown rapidly over the last sixty years, from about 350 million in 1947 to approximately 1.16 billion today. The statistic shows the population growth in India from 2010 to 2020. In 2020, the population of India grew by about 0.99 percent compared to the previous year. We collected the growth rate data is taken from 1998 to 2022. The growth rate percentage is also taken according to the population. The current population of World in 2022 is 7,953,952,567, a 1% increase from 2021. The current population of India is 1,405,754,557 as of Saturday, May 28, 2022, based on Worldometer website elaboration of the latest United Nations data.

Population growth rate is the percentage change in the size of the population in a year. It is calculated by dividing the number of people added to a population in a year (Natural Increase + Net In-Migration) by the population size at the start of the year.



## 6. b Table data:

SMART CIVILIZATION FOR WASTE MANAGEMENT - Power BI Desktop

File Home Help **Table tools**

Name: total generation

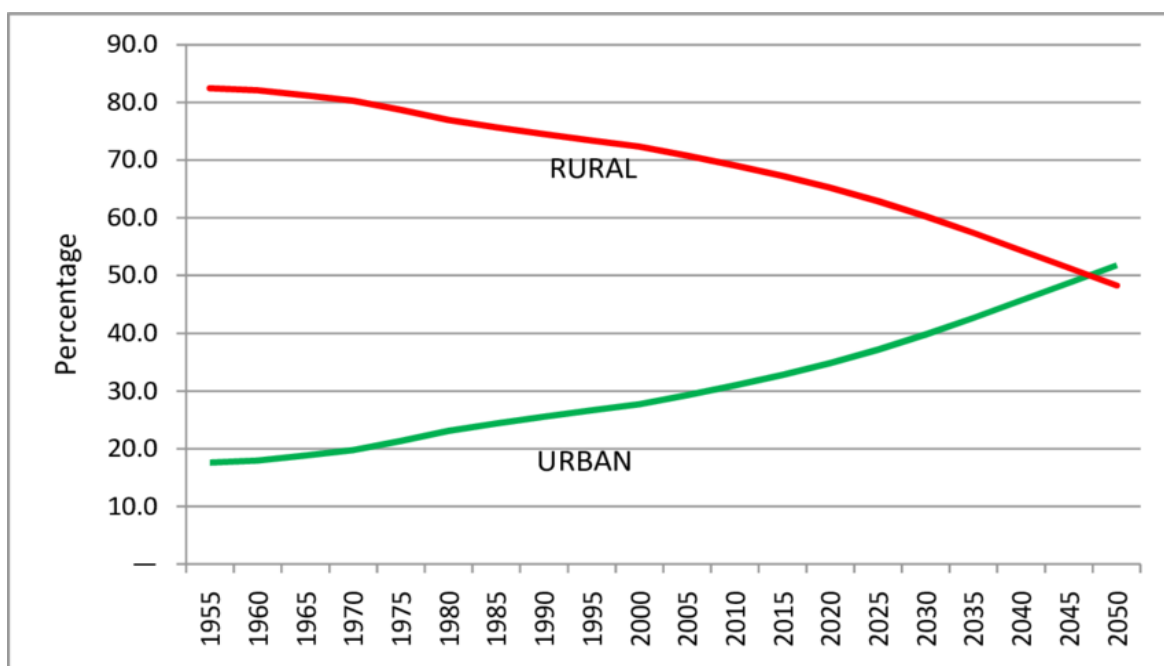
Structure: Mark as date table, Calendars, Manage relationships, Relationships, New measure, Quick measure, New column, New table, Calculations

Area	base population	Growth estimates	yearly waste generation kg/person	waste generated total MT/YEAR
Rural and Urban	1419713452	93.00%	281.24	399280211105
Rural and Urban	1432632844	91.00%	299.04	428414525648
Rural and Urban	1445240013	88.00%	320.4	463054900150
Rural and Urban	1457524553	85.00%	338.2	492934803846
Rural and Urban	1469622007	83.00%	356	523185434440
Rural and Urban	1481525945	81.00%	370.24	548520165917
Rural and Urban	1492933695	77.00%	384.48	574003147010
Rural and Urban	1503981404	74.00%	402.28	605021639293
Rural and Urban	1514810070	72.00%	412.96	625555966647
Rural and Urban	1525262260	69.00%	420.08	640732170107
Rural and Urban	1535328991	66.00%	434.32	666824087257
Rural and Urban	1545001563	63.00%	448.56	693025901270

Fig 6.b: Table Data

### 6.b.1 Description:

Rural and Urban Population is combined to find the growth rate in a country. Rural Population is decreasing rapidly and increasing Urban Population.



## 6.b India Population Growth from 1998 to 2028:

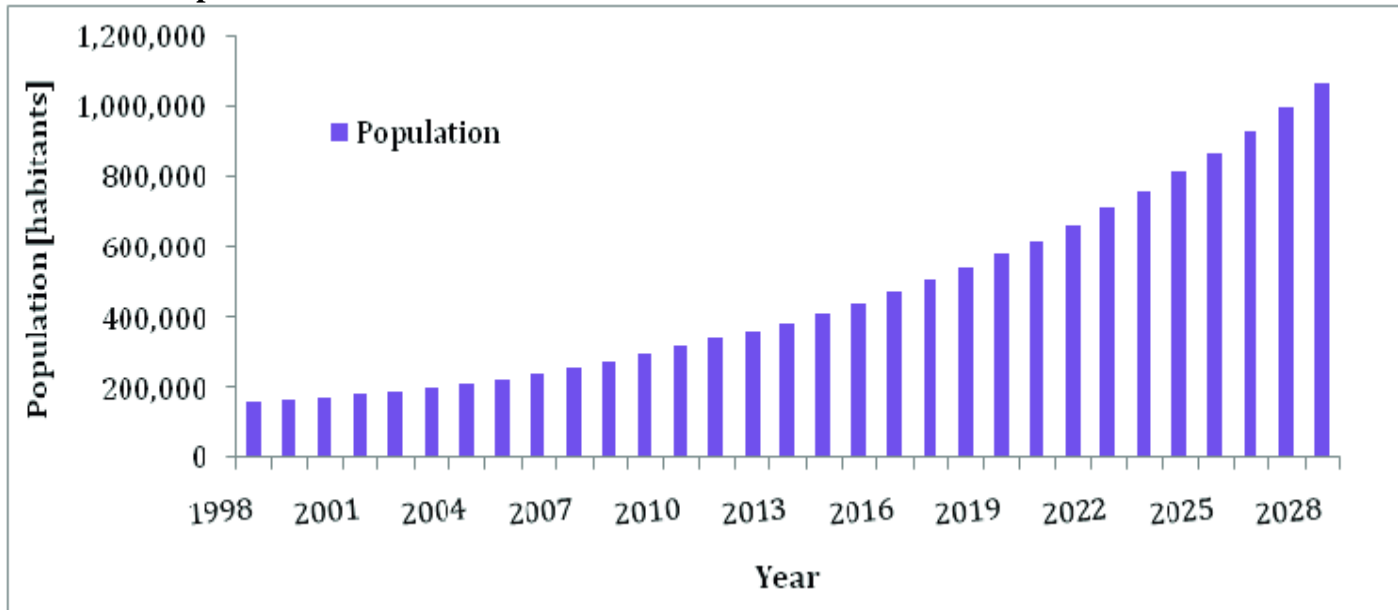


Fig 6.c: Population Growth.

## 6.MS Excel file in Smart Civilization:

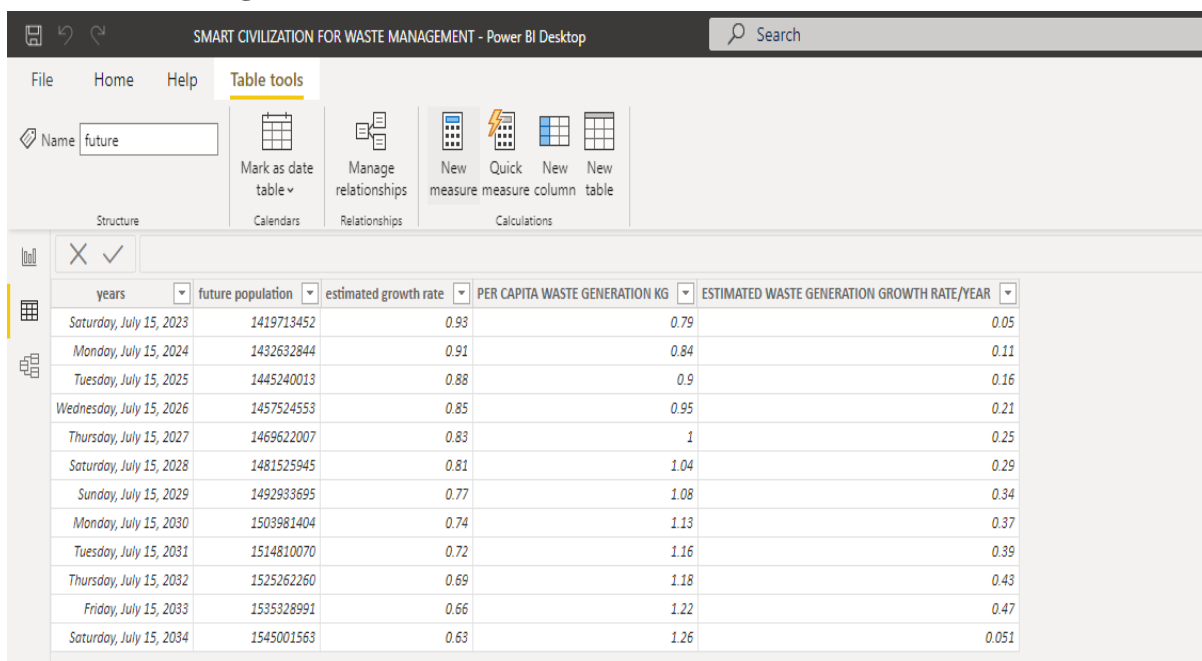
MS Excel tutorial provides basic and advanced concepts of Excel. Our Excel tutorial is designed for beginners and professionals by keeping their requirements in mind.

Microsoft Excel is a computer application program written by Microsoft. It mainly comprises tabs, groups of commands, and worksheets. It stores the data in tabular form and allows the users to perform manipulation operations on them.

## SMART CIVILIZATION MS EXCEL FILE:

1						Area
2						Rural and Urban
3	years	future population	estimated growth r	PER CAPITA WAST	ESTIMATED WASTE GENERATION GROWTH RATE/YEAR	Rural and Urban
4	15-07-2023	1419713452	93.00%	0.79	0.05	Rural and Urban
5	15-07-2024	1432632844	91.00%	0.84	0.11	Rural and Urban
6	15-07-2025	1445240013	88.00%	0.9	0.16	Rural and Urban
7	15-07-2026	1457524553	85.00%	0.95	0.21	Rural and Urban
8	15-07-2027	1469622007	83.00%	1	0.25	Rural and Urban
9	15-07-2028	1481525945	81.00%	1.04	0.29	Rural and Urban
10	15-07-2029	1492933695	77.00%	1.08	0.34	Rural and Urban
11	15-07-2030	1503981404	74.00%	1.13	0.37	Rural and Urban
12	15-07-2031	1514810070	72.00%	1.16	0.39	Rural and Urban
13	15-07-2032	1525262260	69.00%	1.18	0.43	Rural and Urban
14	15-07-2033	1535328991	66.00%	1.22	0.47	Rural and Urban
15	15-07-2034	1545001563	63.00%	1.26	0.051	MSW generation
16	Population growth rate = ((Natural Increase + Net in Migration)/Starting population)) * 100					in small towns to
17						
18						
19	<b>Population Growth and Population Decrease : Formula</b>					
20						
21	The population decrease and growth formula are as below:					0.17kg+0.62kg=0.79kg
22	1. If a constant rate of growth be R% per annum, then population after n years = $P \times (1+R/100)^n$ .					
23	2. if the growth be R% during first year and Q% during second year the population after 2 years = $P \times (1+R/100) \times (1+Q/100)$					
24	3. if the constant decrease in population be R% per annum, then the population after n years = $P \times (1-R/100)^n$ .					
25						
26						
27						

## 6.2 Future Data Image

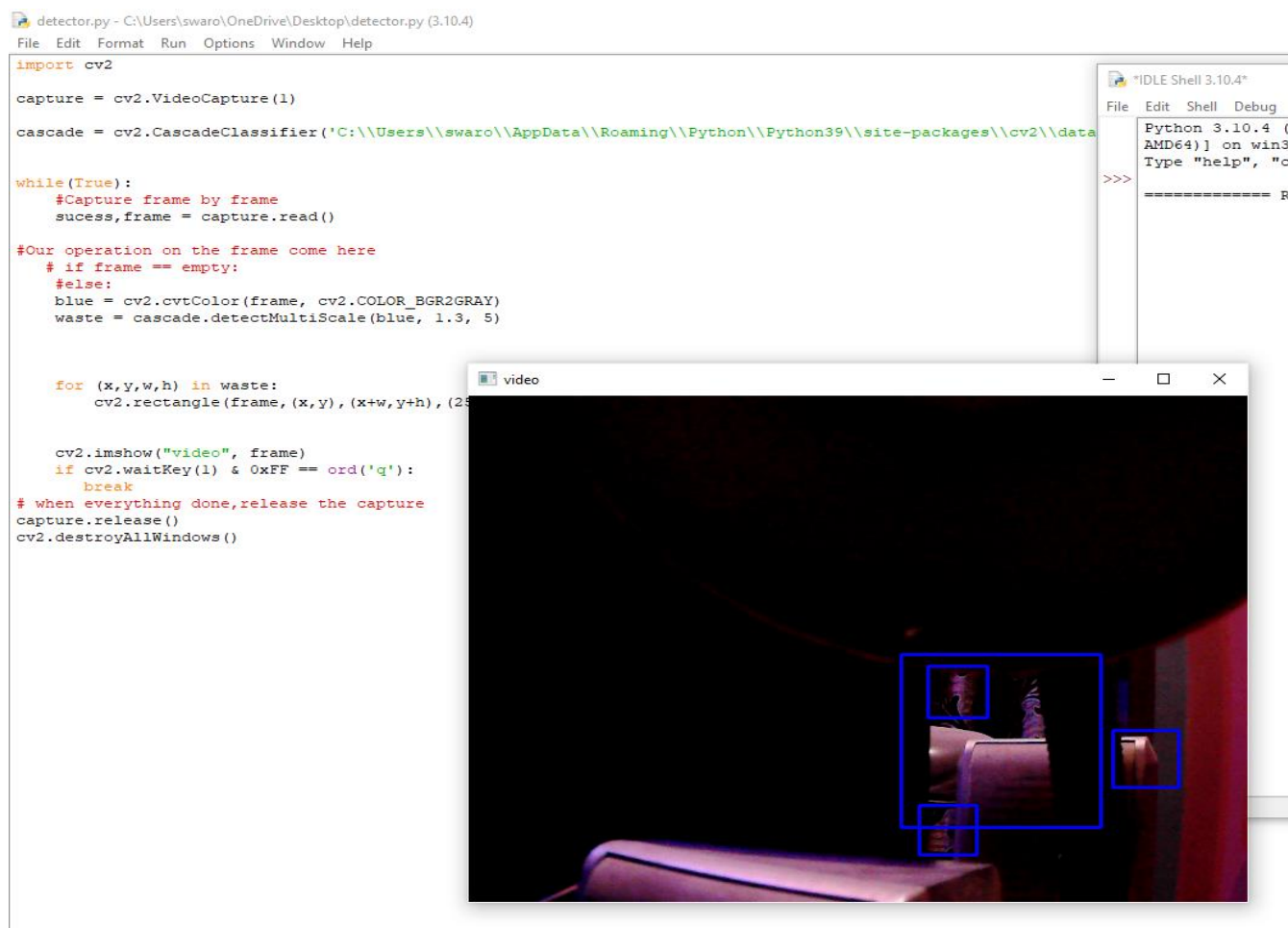


years	future population	estimated growth rate	PER CAPITA WASTE GENERATION KG	ESTIMATED WASTE GENERATION GROWTH RATE/YEAR
Saturday, July 15, 2023	1419713452	0.93	0.79	0.05
Monday, July 15, 2024	1432632844	0.91	0.84	0.11
Tuesday, July 15, 2025	1445240013	0.88	0.9	0.16
Wednesday, July 15, 2026	1457524553	0.85	0.95	0.21
Thursday, July 15, 2027	1469622007	0.83	1	0.25
Saturday, July 15, 2028	1481525945	0.81	1.04	0.29
Sunday, July 15, 2029	1492933695	0.77	1.08	0.34
Monday, July 15, 2030	1503981404	0.74	1.13	0.37
Tuesday, July 15, 2031	1514810070	0.72	1.16	0.39
Thursday, July 15, 2032	1525262260	0.69	1.18	0.43
Friday, July 15, 2033	1535328991	0.66	1.22	0.47
Saturday, July 15, 2034	1545001563	0.63	1.26	0.051

6.3

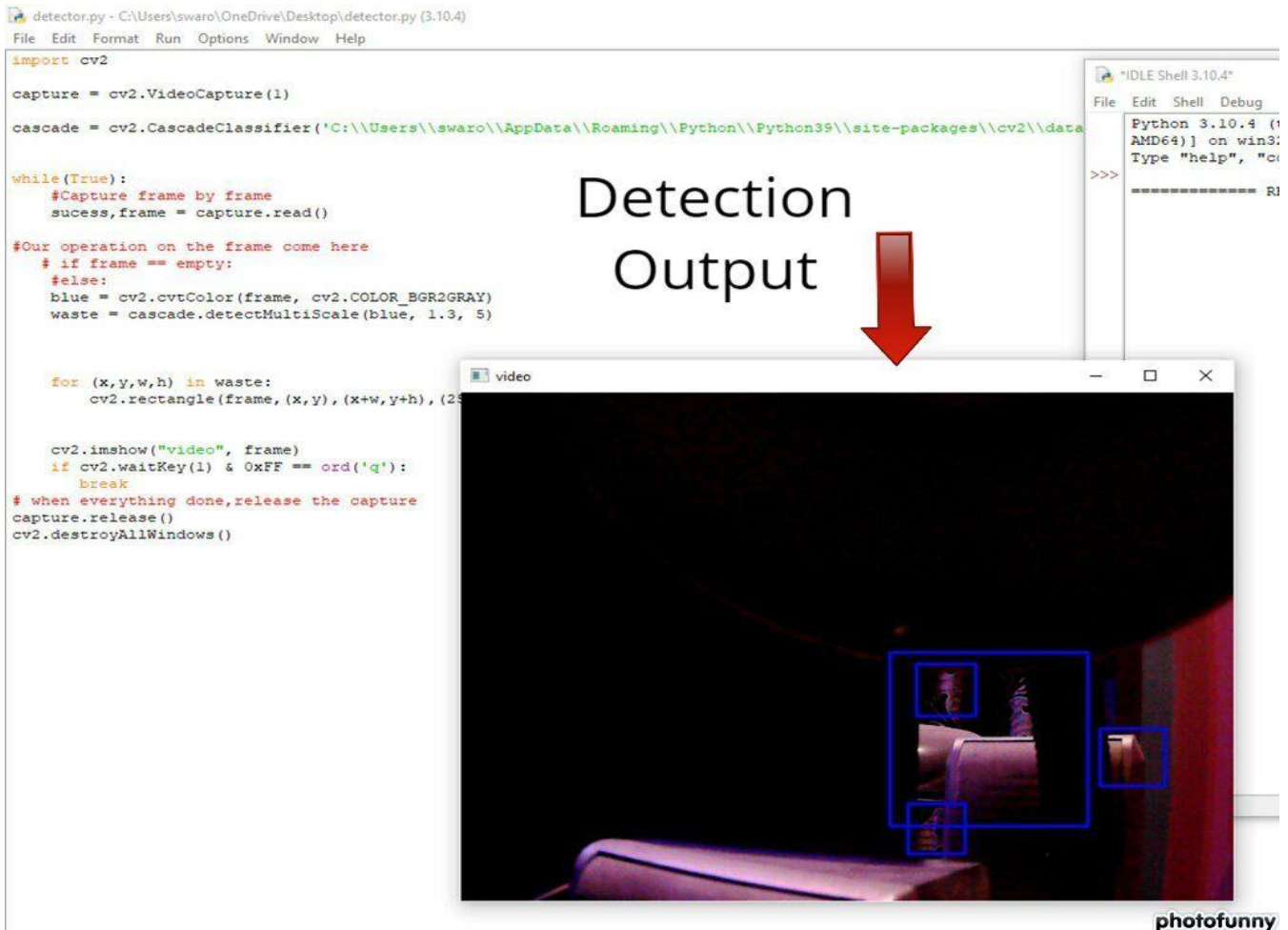
Fig 6.d: Future Data Image.

### 6.3.1 Litter / Garbage Detection Output:



## 6.4 Output of Litter Detection code:

**Note:** In this code to get Output we use an external web camera unless you use you get an cv2 camera error.



6. d Output of Litter Detection code

### 6.4.1 Garbage Detection and PowerBI Dashboard:

In this we will link both Dashboard and Detection code and ensure that both will work simultaneously. We create a URL and connect URL with Dashboard to get the increase of wastage in future and the percentage of wastage present in a particular area.

## 6.5 LitterDetection.png

- Identify Litter on Roads easily.
- Reduce the wastage on roads.

## **Chapter 7**

### **CONCLUSION AND FUTURE ENHANCEMENTS**

Litter is associated with many social, environmental, and economical issues. It can also be a source of safety hazards for road users. The issue of litter is a complex matter caused by various reasons which do not have a single solution. Removal methods focus on collecting the existing litter which can be very effective on a short-term basis but they are usually costly. Furthermore, roadside litter is a growing problem due to the increase in VMT and contributing factors such as the emergence of the COVID-19 pandemic. Therefore, people are littering much faster than DOTs and volunteer programs can pick it up. Preventive methods, on the other hand, following a longer-term approach that might even take up to a generation to be completely effective. A long-term effort for mitigating this problem must include all the discussed methods to gain maximum success. To change the littering behaviors, engaging communities, continuous educational programs, encouraging volunteer works, improving the design of the trash receptacles, revising legislation, and establishing more effective enforcement methods are the major effective strategies. In terms of public awareness, it is also very important to choose the proper methods to maximize the audience and send the right message to the right group. Social media, especially if it is accompanied by celebrities' endorsement is expected to be a strong tool. Although a change in littering attitude is fundamental for any litter abatement program, the role of producers should not be ignored either. The drastic surge in the amount of plastic litter in the last 50 years is a direct result of the increase of the products, and producers' choice of packaging material. Governments need to encourage, or even enforce industries to think and act with a more sustainable approach.

#### **FUTURE ENHANCEMENTS:**

As we added the dashboard for the garbage or Litter detection for getting the Litter percentage. In Future we can also give the alert for an area if the percentage in the dashboard shows more that helps in easy reducing of litter. The alert can be intimating to police or government, Arrange an sound machine in garbage office that helps the officers to clear the litter easily from that area.

## 8. REFERENCES

### General

1. Q. Fang, C. Xu, J. Sang, M. S. Hossain, and A. Ghonim, “Folksonomy-based visual ontology construction and its applications,” *IEEE Transactions on Multimedia*, vol. 18, no. 4, pp. 702–713, 2016.  
View at: [Publisher Site](#) | [Google Scholar](#)
2. H. Rezatofighi, N. Tsoi, J. Gwak, A. Sadeghian, I. Reid, and S. Savarese, “Generalized intersection over union: a metric and a loss for bounding box regression,” in *Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 658–666, Long Beach, CA, USA, June 2019.  
View at: [Google Scholar](#)
3. Z. Zheng, P. Wang, W. Liu, and D. Ren, “Distance-IoU loss: faster and better learning for bounding box regression,” *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 7, pp. 12993–13000, 2020.  
View at: [Publisher Site](#) | [Google Scholar](#)
4. S. Woo, J. Park, J. Y. Lee, and I. S. Kweon, “CBAM: convolutional block attention module,” in *Proceedings of the ECCV, Munich, Germany, September 2018*.  
View at: [Google Scholar](#)
5. Q. Wang, B. Wu, P. Zhu, P. Li, W. Zuo, and Q. Hu, “ECA-net: efficient channel attention for deep convolutional neural networks,” in *Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11531–11539, Seattle, WA, USA, June 2020.  
View at: [Google Scholar](#) Copyright