ML-BASED DERMAL CELL ANALYSIS FOR SKIN CANCER DETECTION

Minor project-II report submitted
in partial fulfillment of the requirement for award of the degree of

Bachelor of Technology
in
Computer Science and Design

By

NANDYALA NAVEEN REDDY	(21UEDL0020)	**(VTU20906)**
DALLI AKASH REDDY	(21UEDL0008)	**(VTU20933)**
KORRA VAMSHI	(21UEDL0016)	**(VTU19677)**

Under the guidance of
Dr. A. BHAGYALAKSHMI, M.E., Ph.D.,
PROFESSOR

DEPARTMENT OF COMPUTER SCIENCE AND DESIGN
SCHOOL OF COMPUTING

VEL TECH RANGARAJAN Dr. SAGUNTHALA R&D INSTITUTE OF SCIENCE AND TECHNOLOGY

(Deemed to be University Estd u/s 3 of UGC Act, 1956)
Accredited by NAAC with A++ Grade
CHENNAI 600 062, TAMILNADU, INDIA

May, 2024

ML-BASED DERMAL CELL ANALYSIS FOR SKIN CANCER DETECTION

Minor project-II report submitted
in partial fulfillment of the requirement for award of the degree of

Bachelor of Technology
in
Computer Science and Design

By

NANDYALA NAVEEN REDDY	(21UEDL0020)	**(VTU20906)**
DALLI AKASH REDDY	(21UEDL0008)	**(VTU20933)**
KORRA VAMSHI	(21UEDL0016)	**(VTU19677)**

Under the guidance of
Dr. A. BHAGYALAKSHMI, M.E., Ph.D.,
PROFESSOR

DEPARTMENT OF COMPUTER SCIENCE AND DESIGN
SCHOOL OF COMPUTING

VEL TECH RANGARAJAN Dr. SAGUNTHALA R&D INSTITUTE OF SCIENCE AND TECHNOLOGY

(Deemed to be University Estd u/s 3 of UGC Act, 1956)
Accredited by NAAC with A++ Grade
CHENNAI 600 062, TAMILNADU, INDIA

May, 2024

CERTIFICATE

It is certified that the work contained in the project report titled "ML-BASED DERMAL CELL ANALYSIS FOR SKIN CANCER DETECTION" by "N NAVEEN REDDY (21UEDL0020), D AKASH REDDY (21UEDL0008), K VAMSHI (21UEDL0016)" has been carried out under my supervision and that this work has not been submitted elsewhere for a degree.

Signature of Supervisor
Computer Science & Engineering
School of Computing
Vel Tech Rangarajan Dr. Sagunthala R&D
Institute of Science & Technology
May, 2024

Signature of Professor In-charge
Computer Science and Design
School of Computing
Vel Tech Rangarajan Dr. Sagunthala R&D
Institute of Science & Technology
May, 2024

DECLARATION

We declare that this written submission represents our ideas in our own words and where other's ideas or words have been included, we have adequately cited and referenced the original sources. We also declare that we have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in our submission. We understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

NANDYALA NAVEEN REDDY

Date: / /

DALLI AKASH REDDY

Date: / /

KORRA VAMSHI

Date: / /

APPROVAL SHEET

This project report entitled “ML-BASED DERMAL CELL ANALYSIS FOR SKIN CANCER DETECTION” by N NAVEEN REDDY (21UEDL0020), D AKASH REDDY (21UEDL0008), K VAMSHI (21UEDL0016) is approved for the degree of B.Tech in Computer Science and Design.

Examiners **Supervisor**

Dr. A . Bhagyalakshmi, M E., Ph.D.,

Date: / /

Place:

ACKNOWLEDGEMENT

We express our deepest gratitude to our respected **Founder Chancellor and President Col. Prof. Dr. R. RANGARAJAN B.E. (EEE), B.E. (MECH), M.S (AUTO), D.Sc., Foundress President Dr. R. SAGUNTHALA RANGARAJAN M.B.B.S.** Chairperson Managing Trustee and Vice President.

We are very much grateful to our beloved **Vice Chancellor Prof. S. SALIVAHANAN,** for providing us with an environment to complete our project successfully.

We record indebtedness to our **Professor & Dean, Department of Computer Science & Engineering, School of Computing, Dr. V. SRINIVASA RAO, M.Tech., Ph.D.,** for immense care and encouragement towards us throughout the course of this project.

We are thankful to our **Head, Department of Computer Science and Design, Dr. R. PARTHASARATHY, M.E., Ph.D.,** for providing immense support in all our endeavors.

We also take this opportunity to express a deep sense of gratitude to our **Internal Supervisor Dr. A. BHAGYALAKSHMI, M.E., Ph.D.,** for her cordial support, valuable information and guidance, she helped us in completing this project through various stages.

A special thanks to our **Project Coordinators Mr. V. KARTHIKEYAN, M.E.,** for his valuable guidance and support throughout the course of the project.

We thank our department faculty, supporting staff and friends for their help and guidance to complete this project.

NANDYALA NAVEEN REDDY (21UEDL0020)
DALLI AKASH REDDY (21UEDL0008)
KORRA VAMSHI (21UEDL0016)

ABSTRACT

Skin cancer is a prevalent and potentially life-threatening disease with increasing incidence worldwide. Early detection plays a crucial role in successful treatment and improved prognosis for patients. Focuses on the development of an advanced skin cancer detection system leveraging cutting-edge technology, such as artificial intelligence and image processing. The proposed system aims to enhance the accuracy and efficiency of skin cancer diagnosis by analyzing dermatoscopic images. Utilizing a deep learning approach, the model is trained on a diverse dataset to recognize various skin cancer types, including melanoma, basal cell carcinoma, and squamous cell carcinoma. The project follows a systematic approach, beginning with the collection of a diverse dataset of dermatoscopic images representing different skin cancer types. Data preprocessing involves image normalization, enhancement, and segmentation to extract relevant features. Feature extraction is a crucial step to capture essential characteristics of skin lesions, enabling effective differentiation between benign and malignant cases. A comprehensive evaluation of machine learning algorithms is conducted to determine the optimal model for skin cancer classification. Commonly employed algorithms, including Support Vector Machine (SVM), Random Forest (RF), and Convolutional Neural Networks (CNN), are implemented and fine-tuned. The model's performance is assessed using metrics such as accuracy, sensitivity, specificity, and area under the Receiver Operating Characteristic Curve (ROCC). The significance of a project focused on skin cancer detection using machine learning is paramount in the realm of healthcare and medical research.

Keywords: Accuracy, Convolutional Neural Networks, Dermatoscopic Image, Interpretability, Medical imaging, Melanoma, Squamous Cell Carcinoma, Support Vector Machine, Sensitivity, Specificity.

LIST OF FIGURES

4.1 **System Architecture of Skin Cancer Detection** 12
4.2 **Data Flow Diagram** 13
4.3 **Use Case Diagram** 14
4.4 **Class Diagram** 15
4.5 **Sequence Diagram** 16
4.6 **Activity Diagram** 17

5.1 **Permissions to Dataset** 29
5.2 **Test Image** 35

6.1 **Processed Skin Tissue** 40

8.1 **Plagarism Report** 43

9.1 **Poster Diagram** 48

LIST OF ACRONYMS AND ABBREVIATIONS

CNN	Convolutional Neural Networks
DI	Dermatoscopic Images
ELM	Extreme Learning Machine
IDE	Integrated Development Environment
ML	Machine Learning
PC	Personal Computer
RF	Random Forest
ROCC	Receiver Operating Characteristic Curve
SVM	Support Vector Machine

TABLE OF CONTENTS

Page.No

ABSTRACT **v**

LIST OF FIGURES **vi**

LIST OF ACRONYMS AND ABBREVIATIONS **vii**

1 INTRODUCTION **1**

1.1 Introduction 1

1.2 Aim of the Project 2

1.3 Project Domain 2

1.4 Scope of the Project 2

2 LITERATURE REVIEW **4**

3 PROJECT DESCRIPTION **8**

3.1 Existing System 8

3.2 Proposed System 8

3.3 Feasibility Study 9

3.3.1 Economic Feasibility 9

3.3.2 Technical Feasibility 10

3.3.3 Social Feasibility 10

3.4 System Specification 11

3.4.1 Hardware Specification 11

3.4.2 Software Specification 11

3.4.3 Standards and Policies 11

4 METHODOLOGY **12**

4.1 General Architecture 12

4.2 Design Phase 13

4.2.1 Data Flow Diagram 13

4.2.2 Use Case Diagram 14

4.2.3 Class Diagram 15

4.2.4 Sequence Diagram 16
4.2.5 Activity Diagram 17
4.3 Algorithm & Pseudo Code 19
4.3.1 Algorithm 19
4.3.2 Pseudo Code 19
4.4 Module Description 22
4.4.1 Data Collection and Preprocessing Module 22
4.4.2 Exploratory Data Analysis (EDA) Module 22
4.4.3 Model Development and Training Module 23
4.4.4 Model Evaluation Module 24
4.5 Steps to execute/run/implement the project 24
4.5.1 Install Required Software 24
4.5.2 Install Necessary Libraries 25
4.5.3 Data Collection and Preprocessing 25
4.5.4 Exploratory Data Analysis (EDA) 26
4.5.5 Model Development and Training 26
4.5.6 Model Evaluation 27
4.5.7 Model Interpretation 27

5 IMPLEMENTATION AND TESTING 29
5.1 Input and Output 29
5.1.1 Permissions to Dataset 29
5.1.2 Processing of the Dermal 30
5.2 Testing 30
5.2.1 Unit Testing 30
5.2.2 Integration Testing 32
5.2.3 System Testing 33
5.2.4 Test Result 35

6 RESULTS AND DISCUSSIONS 36
6.1 Efficiency of the Proposed System 36
6.2 Comparison of Existing and Proposed System 36
6.2.1 Existing System 36
6.2.2 Proposed System 37
6.3 Sample Code 37

7 CONCLUSION AND FUTURE ENHANCEMENTS **41**
7.1 Conclusion 41
7.2 Future Enhancements 42

8 PLAGIARISM REPORT **43**

9 SOURCE CODE & POSTER PRESENTATION **44**
9.1 Source Code 44
9.2 Poster Presentation 48

References **49**

Chapter 1

INTRODUCTION

1.1 Introduction

Skin cancer represents a substantial global health challenge, with its incidence on the rise. Early detection is paramount for effective intervention and improved prognosis. In recent years, the intersection of medical imaging and machine learning has emerged as a promising avenue for enhancing diagnostic capabilities. This project delves into the development of a skin cancer detection system using machine learning, specifically tailored to analyze dermatoscopic images.

Leveraging advanced algorithms, this system aims to discriminate between various skin cancer types, such as melanoma, basal cell carcinoma and squamous cell carcinoma with a focus on achieving high accuracy. The integration of machine learning in dermatology holds the potential to not only expedite the diagnostic process but also to contribute to a paradigm shift in healthcare, enabling timely and precise interventions that can significantly impact patient outcomes.

The primary technical goal of skin cancer detection using machine learning is to develop a robust and accurate algorithm that can analyze dermatoscopic images to automatically identify and classify different types of skin lesions, including benign and malignant cases.

The machine learning model aims to learn complex patterns and features from a diverse dataset, enabling it to make precise predictions about the nature of skin abnormalities. This involves implementing advanced image processing techniques, feature extraction methods, and classification algorithms to achieve high sensitivity and specificity in detecting various forms of skin cancer, such as melanoma, basal cell carcinoma, and squamous cell carcinoma.

1.2 Aim of the Project

The aim of the project, the overarching aim of the project on skin cancer detection using machine learning is to pioneer a robust and accurate diagnostic tool for the early and precise identification of skin lesions. Leveraging the power of advanced machine learning algorithms, the project seeks to develop a system capable of analyzing dermatoscopic images and autonomously classifying various types of skin cancer, including melanoma, basal cell carcinoma, and squamous cell carcinoma. The central focus is on enhancing diagnostic accuracy, minimizing false positives and negatives, and ultimately improving patient outcomes through timely interventions. Beyond accuracy, the project aims to create a seamlessly integrated solution, ensuring practicality and user-friendliness for healthcare professionals.

The primary aim of the project on skin cancer detection using machine learning is to harness advanced computational techniques to create a robust and efficient system for the early and accurate identification of skin lesions. The overarching goal is to enhance the diagnostic process, providing healthcare professionals with a reliable tool that minimizes false positives and negatives, ultimately improving patient outcomes.

1.3 Project Domain

The project on skin cancer detection using machine learning operates within the domain of medical imaging and healthcare technology. Specifically, it addresses the critical intersection of machine learning and dermatology. This domain involves the utilization of advanced computational techniques to analyze dermatoscopic images, with the primary objective of developing a sophisticated diagnostic tool for the early detection and classification of skin cancer. Grounded in the urgency of timely intervention, the project navigates the intricacies of medical image analysis, feature extraction, and classification algorithms to create a system capable of autonomously identifying various types of skin lesions, ranging from benign to malignant.

1.4 Scope of the Project

The scope of the project on skin cancer detection using machine learning is vast and encompasses various crucial dimensions in the intersection of healthcare and

technology. The primary focus is on developing an advanced system that utilizes machine learning algorithms to analyze dermatoscopic images for the accurate and early detection of skin cancer. This involves delving into the intricacies of medical image analysis, with an emphasis on feature extraction, pattern recognition, and classification of diverse skin lesions. The project aims to not only distinguish between benign and malignant cases but also to categorize specific skin cancer subtypes, such as melanoma, basal cell carcinoma, and squamous cell Carcinoma.

The essence of the project lies in revolutionizing the landscape of skin cancer detection through the integration of machine learning. At its core, the project seeks to develop a sophisticated and accurate diagnostic tool capable of autonomously analyzing dermatoscopic images to identify and classify various types of skin lesions. The heart of the project beats with a commitment to early detection, aiming to enhance patient outcomes by identifying skin cancer at its incipient stages.

This involves navigating the intricacies of medical image analysis, delving into advanced machine learning algorithms, and optimizing their performance for robust and efficient detection.

Proactive maintenance strategies are pivotal for sustaining the effectiveness and reliability of a skin cancer detection system based on machine learning. Regular model retraining stands as a cornerstone, ensuring the algorithm remains updated with contemporary data, adapting to emerging patterns in dermatoscopic images.

Continuous data monitoring is imperative, allowing for the early identification of any shifts or biases in input data, which is essential for maintaining the model's robustness. Dynamic feature engineering ensures that feature extraction methods evolve with the latest advancements in image processing technology, enabling the system to capture relevant diagnostic information effectively.

Enhancements in interpretability contribute to a deeper understanding of the model's decisions, fostering trust and consistent utilization by healthcare professionals. Scalability considerations, user training, and compliance with regulatory standards round out the proactive measures, ensuring that the skin cancer detection system remains reliable, secure, and aligned with evolving clinical and technological landscapes.

Chapter 2

LITERATURE REVIEW

[1] Imran et al, 2022 present a novel approach to skin cancer detection by leveraging the power of deep learning models. The authors propose a system that combines the decisions of multiple deep learning algorithms to enhance the accuracy and reliability of skin cancer diagnosis. This ensemble method aims to improve detection rates by addressing the limitations of individual models through a collaborative decision-making process. The study utilizes a large dataset of skin lesion images and evaluates the performance of the proposed method against state-of-the-art techniques. The results demonstrate that the combined decision strategy significantly enhances diagnostic performance, suggesting its potential as a robust tool for early skin cancer detection.

[2] Chishti et al, 2023 explore the recent advancements in antenna-based techniques for the detection and monitoring of critical chronic diseases. The paper provides an in-depth analysis of how antenna technology can be applied in the medical field, particularly for chronic disease management. The authors discuss various types of antennas and their applications, including wearable and implantable antennas, and highlight their potential for real-time health monitoring and early disease detection. The review also covers the challenges and future prospects of integrating antenna-based systems in clinical settings. The extensive evaluation and synthesis of current research make this paper a valuable resource for understanding the intersection of antenna technology and healthcare.

[3] Hamza et al, 2024 introduce a high-sensitivity Microscale triple-band biosensor designed using Terahertz Metamaterials (MTMs) for the diagnosis of non-melanoma skin cancer. The proposed biosensor functions as a perfect absorber, enhancing its ability to detect subtle changes in the biological environment, which is crucial for early and accurate cancer diagnosis. The authors detail the design and optimization process of the biosensor, focusing on its sensitivity and specificity in detecting non-melanoma skin cancer. Through rigorous testing and simulation, the study demon-

strates the biosensor's potential for clinical application, offering a promising tool for non-invasive cancer diagnostics.

[4] Mridha et al, 2023 present an interpretable approach to skin cancer classification using an optimized CNN within a smart healthcare system framework. The study emphasizes the importance of interpretability in AI-driven medical diagnostics, ensuring that the classification results are understandable and actionable for healthcare professionals. The authors describe the development and optimization of their CNN model, which aims to provide accurate and reliable skin cancer diagnosis. The performance of the model is evaluated using a comprehensive dataset of skin lesion images, demonstrating high accuracy and robustness. Additionally, the paper discusses the integration of the CNN-based diagnostic tool into a smart healthcare system, highlighting its potential to enhance early detection and treatment of skin cancer.

[5] Ashraf et al, 2020 propose a novel framework for skin cancer detection that utilizes a Region Of Interest (ROI) based approach combined with transfer learning techniques. The framework is designed to enhance the accuracy of skin cancer diagnosis by focusing on critical regions within skin lesion images, thereby improving the performance of the deep learning models. The authors employ transfer learning to leverage pre-trained neural networks, reducing the computational resources and training time required while maintaining high accuracy. The study demonstrates the effectiveness of this approach through extensive experiments on a well-known dataset, showing significant improvements in detection performance compared to traditional methods.

[6] Schiavoni et al, 2023 present a microwave reflectometry sensing system designed for low-cost, in-vivo skin cancer diagnostics. The proposed system utilizes microwave reflectometry to detect skin cancer non-invasively by analyzing the electromagnetic properties of skin tissues. The authors detail the design and implementation of the sensing system, emphasizing its affordability and potential for widespread clinical use. Through experimental validation, the system demonstrates high sensitivity and specificity in distinguishing between healthy and cancerous tissues. The study highlights the practicality and effectiveness of microwave-based diagnostic tools in providing accessible and reliable skin cancer detection.

[7] Abuzaghleh, Barkana, and Faezipour 2015 present a noninvasive, real-time automated system for analyzing skin lesions to aid in the early detection and prevention of melanoma. The system utilizes advanced imaging and analysis techniques to provide rapid and accurate assessment of skin lesions, helping healthcare professionals identify potentially cancerous growths at an early stage. The paper discusses the design and implementation of the system, including its imaging components and algorithmic analysis methods. Through experimental validation, the authors demonstrate the effectiveness of the system in accurately diagnosing melanoma and facilitating timely intervention.

[8] Andreasen et al, 2021 investigate the use of skin electrical resistance as a biomarker for diagnosing and treating breast cancer, particularly in measuring lymphatic regions. The paper explores the potential of skin electrical resistance as a non-invasive diagnostic tool for detecting breast cancer and monitoring its progression. Additionally, it discusses the therapeutic implications of this biomarker, potentially aiding in the development of targeted treatments. The research contributes to the understanding of novel biomarkers for breast cancer management, highlighting the importance of non-invasive and accessible diagnostic techniques.

[9] Riaz et al, 2023 propose a comprehensive joint learning system for the detection of skin cancer. The system integrates multiple learning techniques to improve the accuracy and reliability of skin cancer detection. By combining different types of data and learning algorithms, including deep learning and machine learning approaches, the system aims to enhance the performance of existing diagnostic methods. The paper presents the design and implementation of the joint learning system, along with experimental results demonstrating its effectiveness in detecting various types of skin cancer lesions.

[10] Adegun and Viriri, 2020 present an automated framework for detecting and classifying skin lesions in dermoscopy images using a Fully Convolutional Network (FCN) integrated with DenseNet architecture. This approach leverages the strengths of both FCNs and DenseNet to achieve high accuracy in segmenting and identifying different types of skin lesions. The framework is designed to assist dermatologists by providing reliable and fast diagnostic tools, potentially improving early detec-

tion and treatment of skin cancer. The authors demonstrate the effectiveness of their method through extensive experiments, showing significant improvements over existing techniques.

Chapter 3

PROJECT DESCRIPTION

3.1 Existing System

Deployment of skin cancer detection using machine learning have showcased advancements in the field of medical image analysis. These systems leverage sophisticated algorithms to analyze dermatoscopic images for the early identification and classification of skin lesions. Notable approaches include the application of deep learning techniques, such as CNN, for feature extraction and pattern recognition.

Many existing systems focus on multi-class classification, distinguishing between benign and malignant lesions and categorizing specific subtypes, including melanoma, basal cell carcinoma, and squamous cell carcinoma. The integration of large and diverse datasets is a common practice to enhance the generalizability of these systems across different skin types and clinical scenarios.

Interpretability features have gained attention to provide insights into the decision-making process of machine learning models, fostering trust among healthcare professionals. These existing systems have shown promising results, ongoing research strives to address challenges such as limited interpretability, diverse dataset representation, and seamless integration into clinical workflows, aiming to further improve the accuracy and practicality of skin cancer detection using machine learning. It's essential to consult the latest literature for the most recent developments and advancements in this rapidly evolving field.

3.2 Proposed System

The proposed system for skin cancer detection using machine learning aims to usher in a new era of precision and efficiency in dermatological diagnostics. Employing cutting-edge deep learning architectures, such as CNN, the system focuses on extracting intricate features from dermatoscopic images, enhancing its ability to discriminate between benign and malignant lesions.

The algorithm for skin cancer detection using machine learning is a systematic process designed to effectively analyze Dermatoscopic Images and distinguish between benign and malignant skin lesions. It commences with the collection of a diverse and comprehensive dataset, containing images representing various skin types and lesion characteristics. Following data preprocessing steps, such as normalization and resizing, the dataset is divided into training and testing sets.

The model is trained on the labeled dataset, where it learns to differentiate between benign and malignant lesions. Subsequently, the algorithm undergoes validation to fine-tune parameters and prevent overfitting, followed by testing on an independent dataset to evaluate its real-world performance. Interpretability features, such as Grad-CAM, may be integrated to provide visual explanations of the model's decisions.

Post-processing steps, including threshold adjustments, refine the model's outputs. The algorithm culminates in the integration of the developed model into the clinical workflow, ensuring that healthcare professionals can seamlessly leverage its capabilities for accurate and timely skin cancer diagnosis. Continuous monitoring and updating mechanisms guarantee the adaptability of the algorithm to evolving skin cancer patterns, reinforcing its reliability in clinical practice.

3.3 Feasibility Study

The feasibility study indicates that developing a machine learning-based skin cancer detection system is technically, economically, and social. With the availability of high-quality datasets, advanced machine learning frameworks, and cloud-based infrastructure, the project can be successfully implemented. The potential benefits in terms of improved patient outcomes and healthcare savings make this project a worthwhile investment. Adhering to regulatory standards and ensuring ethical use of data will be crucial for the project's success and acceptance in the healthcare community.

3.3.1 Economic Feasibility

The economic feasibility of employing machine learning for skin cancer detection necessitates a thorough examination of various cost and benefit factors. Initial expenses encompass recruitment of skilled personnel, data procurement, and

infrastructure setup for model development. Furthermore, the labor-intensive task of gathering and annotating a sizable dataset of skin images demands considerable investment. Substantial computational resources are also required for training and validating the machine learning model, alongside the time and effort needed for fine-tuning and regulatory approval processes. However, the potential market for such a solution is significant, given the prevalence of skin cancer. Cost savings may be realized through earlier detection, reduced biopsy rates, and improved patient outcomes. Nonetheless, ongoing expenses for deployment, maintenance, and ethical considerations must be factored into the equation. A comprehensive feasibility analysis, weighing these factors, is indispensable for informed decision-making regarding economic viability.

3.3.2 Technical Feasibility

The technical feasibility of employing machine learning for skin cancer detection entails a comprehensive examination of various technical aspects. Firstly, acquiring a diverse and annotated dataset of skin images is crucial for training robust machine learning models. Collaboration with dermatologists or medical institutions may facilitate access to high-quality data. Secondly, selecting appropriate machine learning algorithms and architectures that can effectively learn from complex visual patterns in skin images is essential. CNN have shown promising results in this domain due to their ability to automatically extract relevant features from images.

3.3.3 Social Feasibility

The social feasibility of the project focuses on whether people will accept and benefit from it. We need to think about how regular folks and doctors will feel about using this technology. It's crucial that the project respects privacy and keeps personal information safe. We want to make sure that doctors and patients trust and feel comfortable with the skin cancer detection system. Educating the public about the benefits of early detection and how this technology can help save lives is also important. If people understand and believe in the project, they're more likely to use it and share its benefits with others. So, in simple terms, social feasibility is about making sure everyone feels good about and benefits from using this new way of detecting skin cancer.

3.4 System Specification

3.4.1 Hardware Specification

A computer with

- Processor : Pentium IV or higher.
- RAM : 2GB.
- Space on Disk : minimum 512 MB.

3.4.2 Software Specification

- Python, Google Colab and Arduino IDE.
- Operation systems Supported : Windows 7, Windows 8, Windows 10, Windows 11, Mac M1, Mac M2.

3.4.3 Standards and Policies

Google Colab

Google Colab is a type of command line interface which explicitly deals with the ML(MachineLearning) modules. And navigator is available in all the Windows,Linux and MacOS.The Google Colab has many number of IDE's which make the coding easier. The UI can also be implemented in python.

Standard Used: ISO/IEC 27001

CNN Algorithm

A CNN is a type of artificial neural network designed specifically for processing and analyzing visual data. It has proven highly effective in tasks such as image classification, object detection, and image generation.

Standard Used: ISO/IEC 27001

Chapter 4

METHODOLOGY

4.1 General Architecture

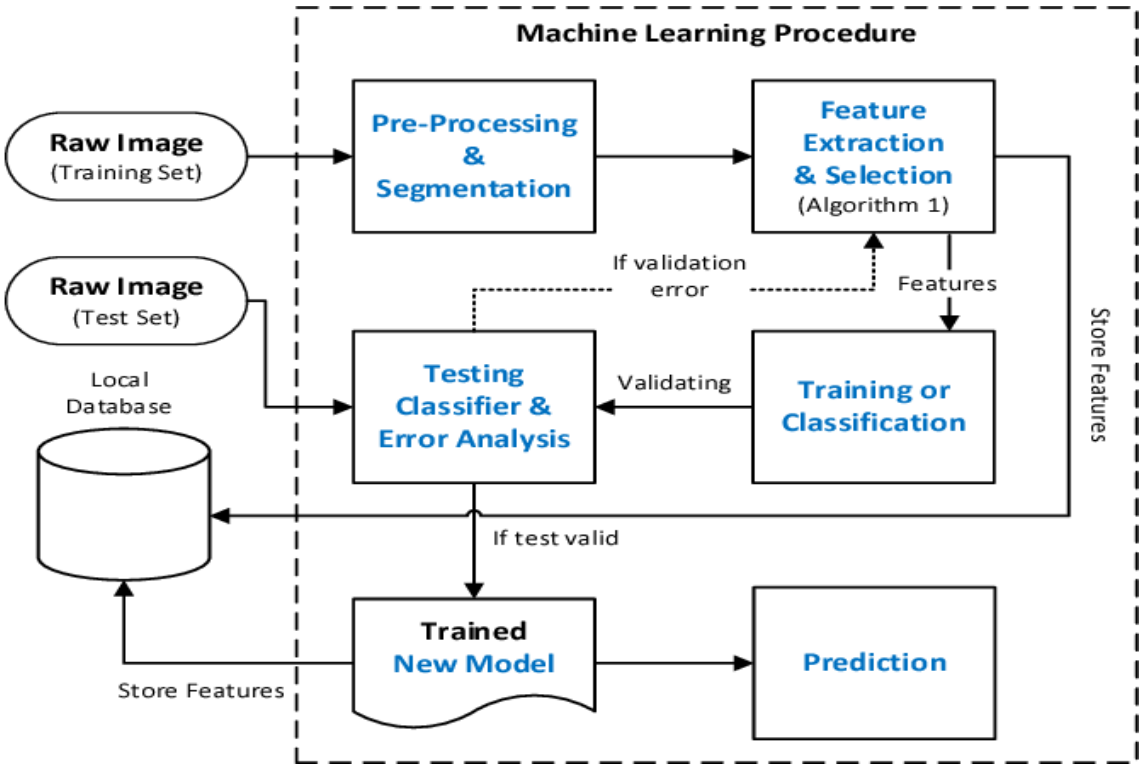


Figure 4.1: **System Architecture of Skin Cancer Detection**

The Figure 4.1 shows system architecture is designed with a comprehensive approach for skin cancer detection. After that, clean the data to obtain the required information. Next, perform data preprocessing, which involves cleaning, transforming, and organizing the raw data to improve its quality. Then visualize the collected data using graphs, which makes it easy to interpret and reveal patterns for efficient decision-making. Also quantify the degree of linear relationship between two variables in the data using the correlation coefficient, which ranges from -1 to 1.

4.2 Design Phase

4.2.1 Data Flow Diagram

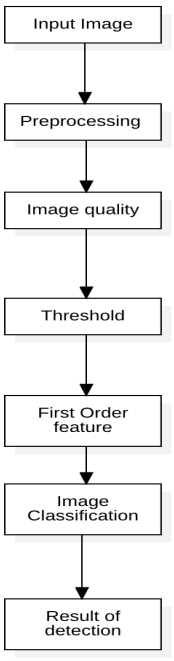


Figure 4.2: **Data Flow Diagram**

The Figure 4.2 shows the Data Flow Diagram to understand systematic progression of information in the skin cancer detection system leverages advanced machine learning techniques to analyze dermoscopic images and provide accurate diagnostic results. The process begins with the acquisition of a high-quality dermoscopic image of the skin lesion, which is then input into the system. This image undergoes several preprocessing steps to enhance its quality, including normalization of intensity values, noise reduction to remove artifacts, and contrast enhancement to highlight critical features of the lesion. After preprocessing, the system performs a quality assessment to ensure the image is suitable for analysis. If the image meets the quality standards, it is subjected to thresholding, a technique that segments the image to isolate regions of interest, such as potential skin lesions. The system then extracts first-order features from the segmented image. These features, which include statistical measures like mean, variance, skewness, and kurtosis, provide essential information about the texture and structure of the lesion.

4.2.2 Use Case Diagram

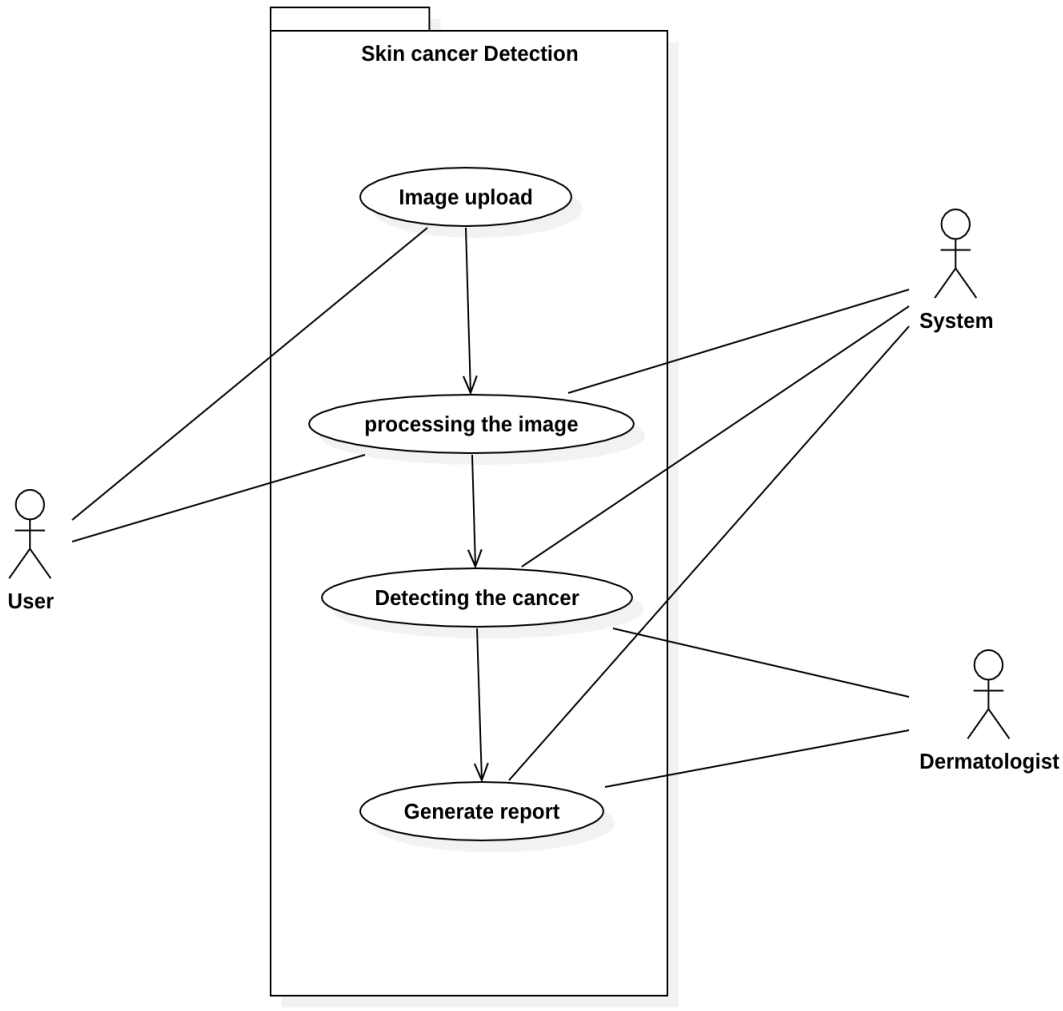


Figure 4.3: **Use Case Diagram**

The Figure 4.3 shows the use case diagram to illustrate the interactions between modules. The system initiates with a user uploading an image of a concerning mole or lesion. This image is then analyzed by the system, likely using image recognition and feature extraction techniques. Based on this analysis, the system determines the potential presence of cancer and generates a report. This report typically includes a risk score and may suggest consulting a dermatologist. While the system offers an initial assessment, the use case diagram highlights the potential involvement of a dermatologist for final diagnosis, who would review the image and the system's report to provide a conclusive evaluation.

4.2.3 Class Diagram

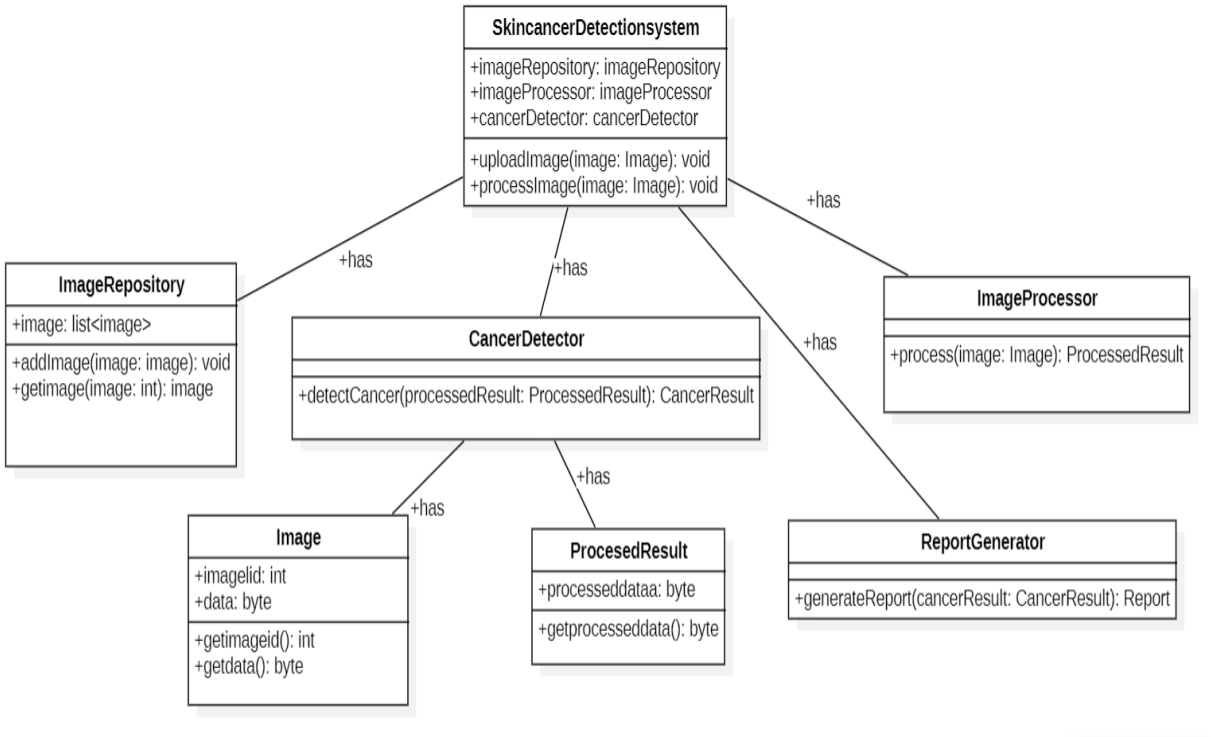


Figure 4.4: **Class Diagram**

The Figure 4.4 shows the class diagram which outlines the fundamental components of the skin cancer detection. A user uploads an image of a suspicious mole or lesion through the upload image function. The image data is likely stored in the image repository. The image processor component receives the image and performs processing tasks on it. This might involve image enhancement or feature extraction techniques to prepare the image for analysis by the cancer detection component. The cancer detector component takes the processed image data from the image processor and analyzes it to determine the likelihood of cancer. It generates a cancer result object that contains details about the analysis, such as the cancer type (if detected) and a confidence level. The report generator component takes the cancer result object and uses it to generate a report (presumably in a human-readable format like a PDF or text document) containing details about the analysis, including the cancer type and confidence level retrieved from the cancer result object.

4.2.4 Sequence Diagram

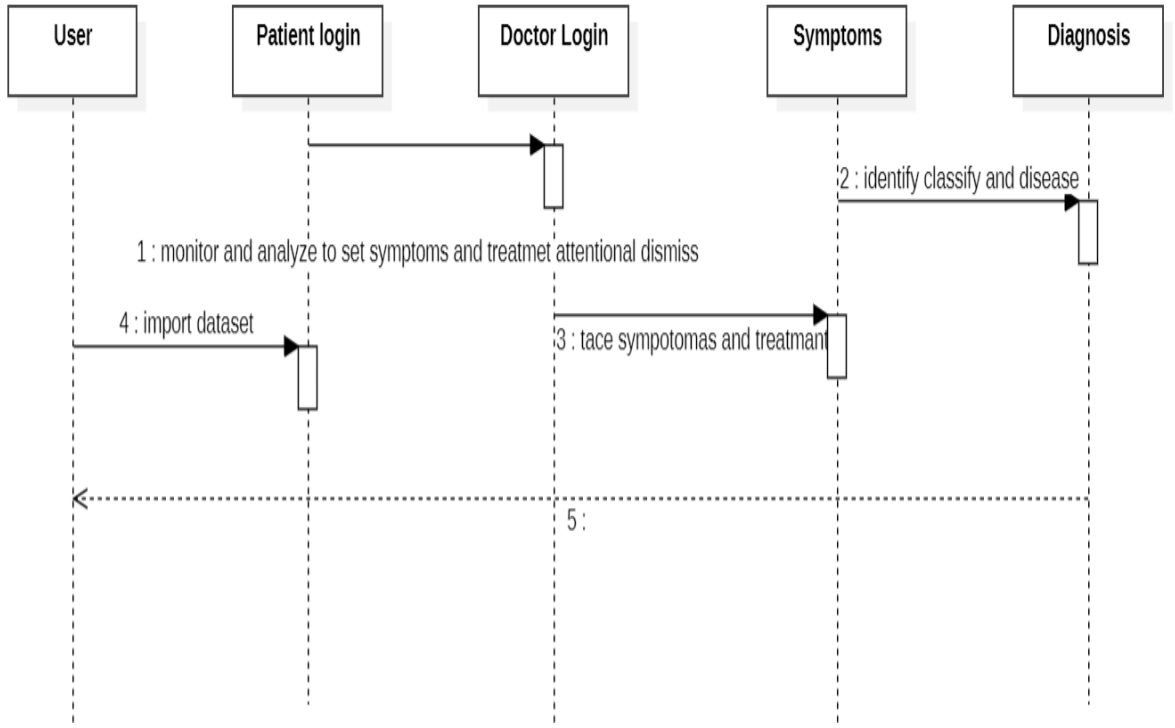


Figure 4.5: **Sequence Diagram**

The Figure 4.5 shows the Sequence Diagram which illustrates the dynamic interactions within the system. Sequence diagram represents the passage of messages. The process commences with the system importing a dataset, most likely containing patient information and past illness records. The system then analyzes and interprets symptoms with the intent to recognize and categorize potential illnesses. The diagram doesn't clarify whether this analysis happens in real-time based on patient monitoring or retrospectively based on the imported data. Depending on the analysis outcome, the system prioritizes the identified diseases. This could involve focusing on diseases with high probability or dismissing less likely ones. If a particular disease is deemed a strong possibility, the system traces the connection between symptoms and known treatments associated with that disease to identify potential options. Finally, the system incorporates the identified disease and its corresponding treatments into its knowledge base. It's unclear from the diagram if this update reflects a diagnosis for a specific patient or if it contributes to the system's overall knowledge for future diagnoses.

4.2.5 Activity Diagram

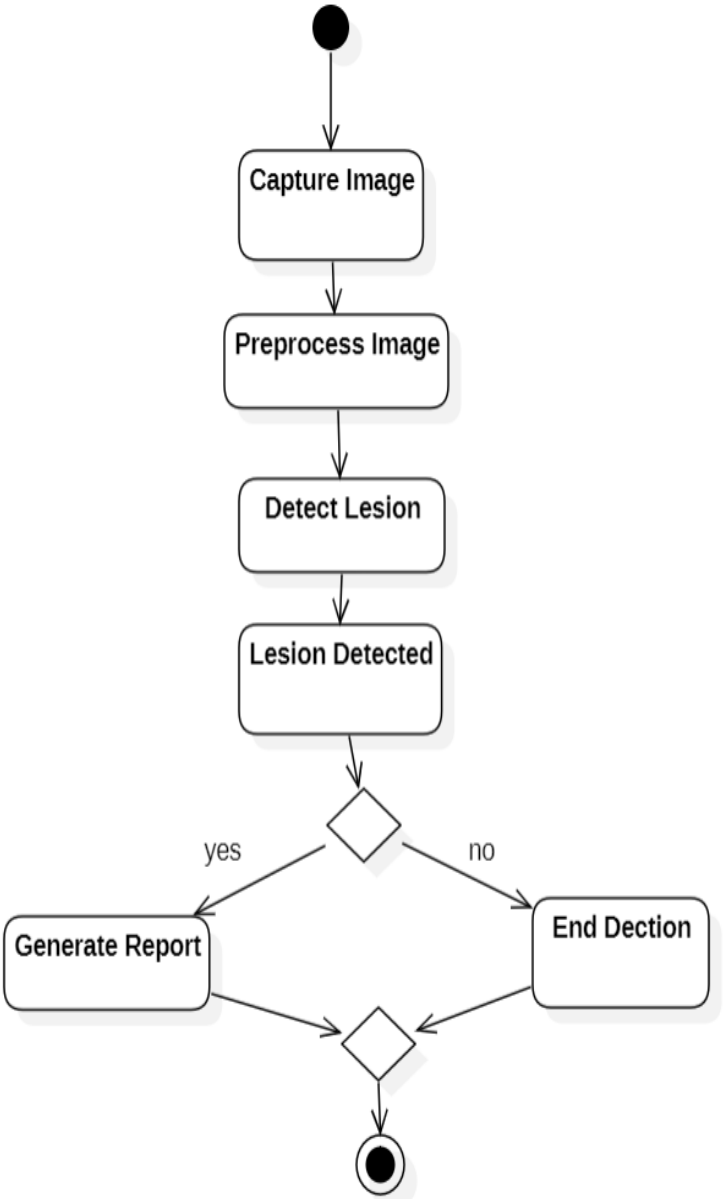


Figure 4.6: **Activity Diagram**

The Figure 4.6 shows the activity diagram you provided offers a step-by-step view of a skin cancer detection system. The process starts with a user capturing an image of a concerning mole or lesion, either through the system's built-in camera or by uploading an existing image. To prepare the image for analysis, a preprocessing step follows. This might involve resizing the image, converting it to grayscale, or applying filters to enhance features crucial for cancer detection. Once preprocessed, the image is analyzed to identify the presence of a lesion. A decision point then checks if a lesion was successfully detected. If a lesion is found, the system generates

a report, likely containing details about the analysis, such as the size, shape, and color of the detected lesion. Conversely, if no lesion is identified, the system concludes the analysis.

4.3 Algorithm & Pseudo Code

4.3.1 Algorithm

Step 1: Start the process.
Step 2: Clean and load the dataset into the code.
Step 3: Generate and train the dataset using Machine Learning Algorithms.
Step 4: Run the unit code.
Step 5: Visualize the efficiency and accuracy of the graph.
Step 6: Stop the process.

4.3.2 Pseudo Code

```

# Import necessary libraries
import pandas as pd
import numpy as np
import seaborn as sns
import plotly.express as px
import matplotlib.pyplot as plt
from sklearn.preprocessing import StandardScaler
from sklearn.model selection import train test split
from sklearn.metrics import accuracy score, confusion matrix
from sklearn.linear model import LogisticRegression
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier, AdaBoostClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.svm import SVC
from xgboost import XGBClassifier

# Load the dataset
df = pd.read csv("/content/drive/Colab Notebooks/Minor 1/waterpotability.csv")

# Data preprocessing and visualization
# ...

# Handle missing values
# ...

# Feature scaling
# ...

# Split the dataset into training and testing sets
# ...

# Model training and evaluation
```

```
34
35 # Logistic Regression
36 model_lr = LogisticRegression()
37 model_lr.fit(x_train, y_train)
38 predlr = model_lr.predict(x_test)
39 accuracylr = accuracyscore(y_test, predlr)
40
41 # Decision Tree
42 model_dt = DecisionTreeClassifier(max_depth=4)
43 model_dt.fit(x_train, y_train)
44 pred_dt = model_dt.predict(x_test)
45 accuracy_dt = accuracy_score(y_test, pred_dt)
46
47 # Random Forest
48 modelrf = RandomForestClassifier()
49 modelrf.fit(xtrain, ytrain)
50 predrf = modelrf.predict(xtest)
51 accuracyrf = accuracyscore(ytest, predrf)
52
53 # K-Nearest Neighour
54 modelknn = K-NearestClassifier(nneighbors=11)
55 modelknn.fit(xtrain, ytrain)
56 predknn = modelknn.predict(xtest)
57 accuracyknn = accuracyscore(ytest, predknn)
58
59 # Support Vector Machine (SVM)
60 modelsvm = SVC(kernel="rbf")
61 modelsvm.fit(xtrain, ytrain)
62 predsvm = modelsvm.predict(xtest)
63 accuracysvm = accuracyscore(ytest, predsvm)
64
65 # AdaBoost
66 modelada = AdaBoostClassifier(nestimators=200, learningrate=0.03)
67 modelada.fit(xtrain, ytrain)
68 predda = modelada.predict(xtest)
69 accuracyada = accuracyscore(ytest, predada)
70
71 # XGBoost
72 modelxgb = XGBClassifier(nestimators=100, learningrate=0.04)
73 modelxgb.fit(xtrain, ytrain)
74 predxgb = modelxgb.predict(xtest)
75 accuracyxgb = accuracyscore(ytest, predxgb)
76
77 # Create a dataframe to store model accuracies
78 models = pd.DataFrame({
79     "Model": ["Logistic Regression", "Decision Tree", "Random Forest",
80               "KNN", "SVM", "AdaBoost", "XGBoost"],
81     "Accuracy": [accuracylr, accuracydt, accuracyrf,
82                  accuracyknn, accuracysvm, accuracyada, accuracyxgb]
83 })
```

```
# Logistic Regression
model_lr = LogisticRegression()
model_lr.fit(x_train, y_train)
predlr = model_lr.predict(x_test)
accuracylr = accuracyscore(y_test, predlr)

# Decision Tree
model_dt = DecisionTreeClassifier(max_depth=4)
model_dt.fit(x_train, y_train)
pred_dt = model_dt.predict(x_test)
accuracy_dt = accuracy_score(y_test, pred_dt)

# Random Forest
modelrf = RandomForestClassifier()
modelrf.fit(xtrain, ytrain)
predrf = modelrf.predict(xtest)
accuracyrf = accuracyscore(ytest, predrf)

# K-Nearest Neighour
modelknn = K-NearestClassifier(nneighbors=11)
modelknn.fit(xtrain, ytrain)
predknn = modelknn.predict(xtest)
accuracyknn = accuracyscore(ytest, predknn)

# Visualize model accuracies
# ...

# Display sorted model accuracies
# ...
```

4.4 Module Description

4.4.1 Data Collection and Preprocessing Module

This module handles the acquisition and preparation of dermoscopic image data, ensuring it is ready for training the machine learning models.

Data Acquisition:

- Sources: ISIC Archive, HAM10000, PH2 Database.
- Methods: API access, web scraping (if applicable), and manual downloading.

Data Cleaning:

- Removal of duplicate, irrelevant, or low-quality images.
- Correction of labeling errors.

Data Annotation:

- Labeling images as benign or malignant (or further specific types if multiclass).
- Use of tools like Labelbox or VGG Image Annotator for manual annotation.

Data Augmentation:

Techniques: Rotation, flipping, zooming, and color adjustments.

Data Normalization:

Scaling pixel values to a standard range (e.g., 0-1).

4.4.2 Exploratory Data Analysis (EDA) Module

This module involves analyzing the dataset to understand its characteristics and prepare for model training.

Statistical Analysis:

- Summarizing class distribution
- image dimensions
- pixel intensity values.

Visualization:

- Histograms
- box plots
- sample image displays.

Handling Class Imbalance:

Techniques like SMOTE (Synthetic Minority Over-sampling Technique) or under-sampling.

4.4.3 Model Development and Training Module

This module focuses on building and training the machine learning models.

Model Selection:

- Baseline models: Simple CNN architectures.
- Advanced models: Pre-trained models like VGG16, ResNet50, InceptionV3, EfficientNet. Custom architectures tailored to the problem.

Transfer Learning:

Fine-tuning pre-trained models on the skin cancer dataset.

Data Splitting:

- Dividing data into training
- Validation
- Test sets.

Training:

- Loss Functions: Binary cross-entropy or categorical cross-entropy.
- Optimizers: Adam, with learning rate scheduling.
- Hyperparameter Tuning: Grid search or random search for optimal parameters.

Validation:

K-Fold Cross-Validation to ensure robustness.

4.4.4 Model Evaluation Module

This module evaluates the performance of trained models using various metrics and techniques.

Performance Metrics:

- Accuracy, Precision, Recall, F1-Score.
- ROC-AUC for measuring classification effectiveness.

Confusion Matrix:

Visualizing true vs Predicted classifications.

Model Interpretability:

- Grad-CAM: Visualizing important regions in the image.
- LIME: Local explanations for individual predictions.

4.5 Steps to execute/run/implement the project

4.5.1 Install Required Software

- Python (latest stable version)
- Jupyter Notebook or an Integrated Development Environment (IDE) like PyCharm or VSCode

4.5.2 Install Necessary Libraries

- TensorFlow/Keras, PyTorch (for deep learning)
- NumPy, Pandas (for data manipulation)
- OpenCV, PIL (for image processing)
- Matplotlib, Seaborn (for data visualization)
- Scikit-learn (for machine learning utilities)
- Flask/Django (for deployment as a web application)
- TensorFlow Lite (for mobile deployment)

4.5.3 Data Collection and Preprocessing

Data Preprocessing:

- Load images and labels using Python libraries.
- Clean and annotate the dataset.
- Apply data augmentation techniques.

```
import os
import cv2
import numpy as np
from sklearn.model_selection import train_test_split

# Example function to load images
def load_images_from_folder(folder):
    images = []
    labels = []
    for filename in os.listdir(folder):
        img = cv2.imread(os.path.join(folder, filename))
        if img is not None:
            images.append(img)
            labels.append(filename.split('_')[0])  # assuming labels are part of the filename
    return images, labels

# Example data augmentation using Keras
from tensorflow.keras.preprocessing.image import ImageDataGenerator

datagen = ImageDataGenerator(
    rescale=1./255,
    rotation_range=40,
    width_shift_range=0.2,
```

```
    height_shift_range=0.2,
    shear_range=0.2,
    zoom_range=0.2,
    horizontal_flip=True,
    fill_mode='nearest'
)

# Example: loading and splitting data
images, labels = load_images_from_folder('data_folder')
train_images, test_images, train_labels, test_labels = train_test_split(images, labels, test_size
    =0.2, random_state=42)
```

4.5.4 Exploratory Data Analysis (EDA)

Handle Class Imbalance:

- Exploratory Data Analysis (EDA)

```
import matplotlib.pyplot as plt

# Example: Plotting image samples
plt.figure(figsize=(10, 10))
for i in range(9):
    plt.subplot(3, 3, i+1)
    plt.imshow(train_images[i])
    plt.title(train_labels[i])
    plt.axis('off')
plt.show()
```

4.5.5 Model Development and Training

Compile and Train the Model:

- Split the data into training, validation, and test sets.
- Train the model using the training and validation sets.

```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Conv2D, MaxPooling2D, Flatten, Dense, Dropout

# Example: Define a simple CNN model
model = Sequential([
    Conv2D(32, (3, 3), activation='relu', input_shape=(224, 224, 3)),
```

```
    MaxPooling2D(pool_size=(2, 2)),
    Conv2D(64, (3, 3), activation='relu'),
    MaxPooling2D(pool_size=(2, 2)),
    Flatten(),
    Dense(128, activation='relu'),
    Dropout(0.5),
    Dense(1, activation='sigmoid')  # Use 'softmax' for multi-class classification
])

# Compile the model
model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])

# Train the model
history = model.fit(
    datagen.flow(np.array(train_images), np.array(train_labels), batch_size=32),
    validation_data=(np.array(test_images), np.array(test_labels)),
    epochs=25
)
```

4.5.6 Model Evaluation

Evaluate Model Performance:

- Use metrics such as accuracy, precision, recall, F1-score, and ROC-AUC.

```
from sklearn.metrics import classification_report, confusion_matrix, roc_auc_score

# Example: Evaluate the model
test_loss, test_acc = model.evaluate(np.array(test_images), np.array(test_labels))
print(f'Test Accuracy: {test_acc}')

# Confusion matrix and classification report
predictions = model.predict(np.array(test_images))
pred_labels = [1 if pred > 0.5 else 0 for pred in predictions]

print(confusion_matrix(test_labels, pred_labels))
print(classification_report(test_labels, pred_labels))
print(f'ROC-AUC: {roc_auc_score(test_labels, predictions)}')
```

4.5.7 Model Interpretation

Use Grad-CAM or LIME for Model Interpretability:

```python
import numpy as np
import matplotlib.pyplot as plt
from tensorflow.keras.preprocessing import image
from tensorflow.keras.models import Model
import tensorflow as tf

# Example: Grad-CAM implementation
def make_gradcam_heatmap(img_array, model, last_conv_layer_name, classifier_layer_names):
    grad_model = Model([model.inputs], [model.get_layer(last_conv_layer_name).output, model.output])
    with tf.GradientTape() as tape:
        conv_outputs, predictions = grad_model(img_array)
        loss = predictions[:, 0]

    output = conv_outputs[0]
    grads = tape.gradient(loss, conv_outputs)[0]
    pooled_grads = tf.reduce_mean(grads, axis=(0, 1, 2))

    heatmap = np.dot(output, pooled_grads[..., np.newaxis])
    heatmap = np.squeeze(heatmap)
    heatmap = np.maximum(heatmap, 0) / np.max(heatmap)
    return heatmap

# Load an image
img_path = 'path_to_image'
img = image.load_img(img_path, target_size=(224, 224))
img_array = image.img_to_array(img)
img_array = np.expand_dims(img_array, axis=0)

# Generate Grad-CAM heatmap
heatmap = make_gradcam_heatmap(img_array, model, 'conv2d_2', ['dense', 'dense_1'])
plt.matshow(heatmap)
plt.show()
```

Chapter 5

IMPLEMENTATION AND TESTING

5.1 Input and Output

5.1.1 Permissions to Dataset

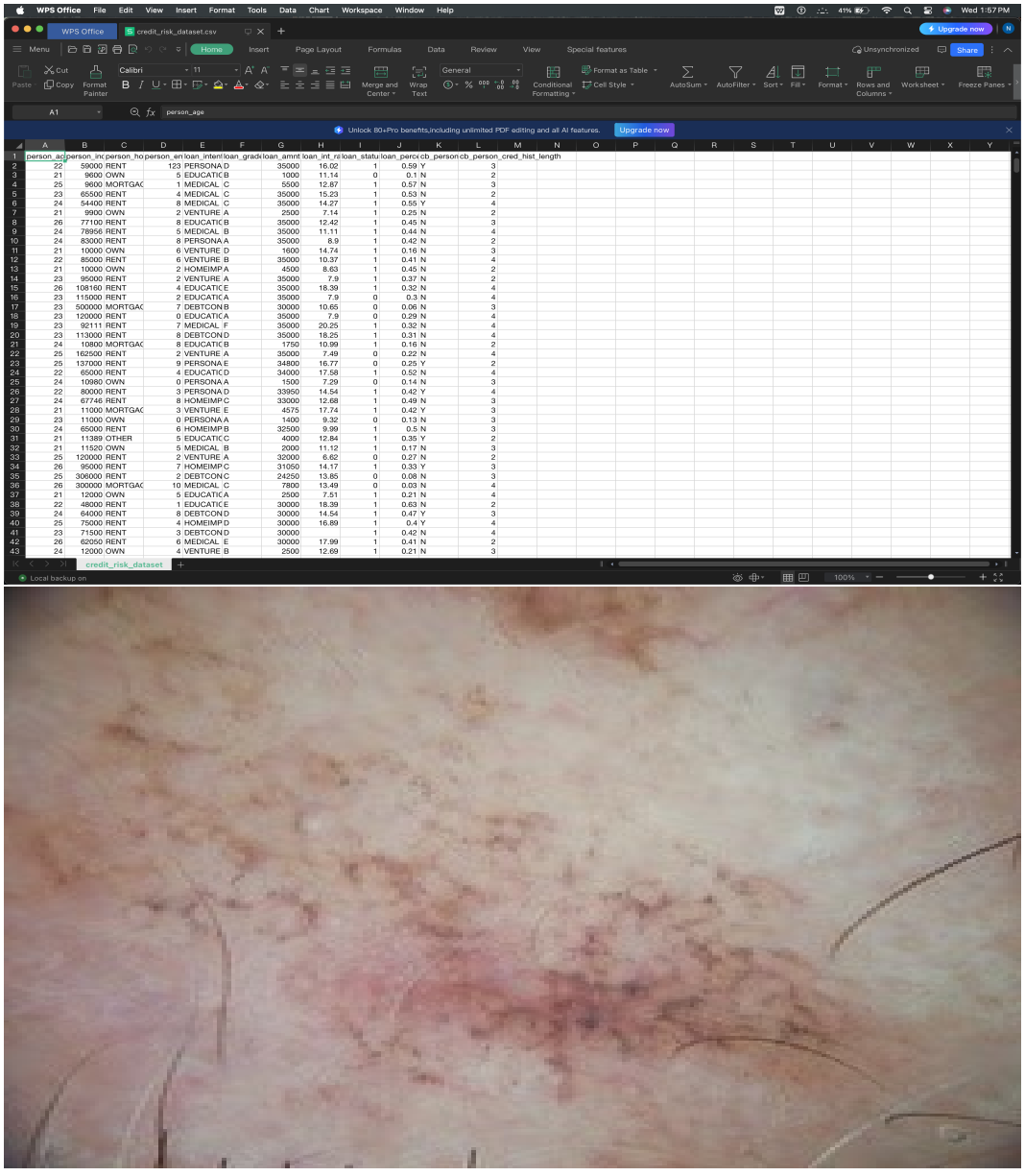


Figure 5.1: **Permissions to Dataset**

The Figure 5.1 depicts the Permissions to Dataset. In this project the dataset consists of 1500 records in the form of excel sheet in csv format as input to train algorithm.

5.1.2 Processing of the Dermal

Here the output is in the form of accuracy where the machine learning algorithms give the accuracy by using the data from the dataset and finally, as an output the accuracies will be obtained so that efficient machine learning algorithm can be taken from the set of algorithms.

5.2 Testing

The foundation of software testing lies in unit testing. Just like examining individual parts of a complex machine to verify their functionality, unit testing focuses on the proper operation of each software unit, such as functions, modules, or classes. Developers write unit tests to validate these units with various inputs, ensuring they produce the expected results. This approach helps catch errors early in development, leading to faster debugging and a more reliable codebase.

5.2.1 Unit Testing

Unit testing is used to ensure that each modular component of the project is working. The smallest unit of the software design is the subject of unit testing. The mentioned project underwent a progression examination of unit testing.

Input

Logicstic Regression

```
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score

# Initialize the Logistic Regression model
model_lr = LogisticRegression()

# Training the model
model_lr.fit(x_train, y_train)

# Make predictions on the test set
```

```
pred_lr = model_lr.predict(x_test)

# Calculate accuracy score
accuracy_score_lr = accuracy_score(y_test, pred_lr)
accuracy_score_lr *= 100  # Multiply by 100 to get percentage

print(f"Accuracy of Logistic Regression Model: {accuracy_score_lr:.2f}%")
```

Random Forest Classifier

```
from sklearn.ensemble import RandomForestClassifier

# Creating model object
model_rf = RandomForestClassifier()
#Training Model RF
model_rf.fit(x_train,y_train)
#Making Prediction
pred_rf = model_rf.predict(x_test)
accuracy_score_rf = accuracy_score(y_test,pred_rf)
accuracy_score_rf*100
cm3 = confusion_matrix(y_test,pred_rf)
cm3
```

Decision Tree

```
from sklearn.tree import DecisionTreeClassifier

# creating the model object
model_dt = DecisionTreeClassifier(max_depth = 4)
#Training of decision tree
model_dt.fit(x_train,y_train)
#Making prediction using Decision Tree
pred_dt = model_dt.predict(x_test)
accuracy_score_dt = accuracy_score(y_test,pred_dt)
accuracy_score_dt*100
cm2 = confusion_matrix(y_test,pred_dt)
cm2
```

K-Nearest Neighbour

```
from sklearn.neighbors import KNeighborsClassifier

#Creating Model object
model_knn = KNeighborsClassifier()

for i in range(4,15):
```

```
    model_knn = KNeighborsClassifier(n_neighbors=i)
    model_knn.fit(x_train, y_train)
    pred_knn = model_knn.predict(x_test)
    accuracy_score_knn = accuracy_score(y_test, pred_knn)
    print(i, accuracy_score_knn)
model_knn = KNeighborsClassifier(n_neighbors=11)
model_knn.fit(x_train, y_train)
pred_knn = model_knn.predict(x_test)
accuracy_score_knn = accuracy_score(y_test, pred_knn)
print(accuracy_score_knn*100)
```

Test result

The unit testing findings were encouraging.

5.2.2 Integration Testing

Integration testing is a methodical methodology for building the software architecture while also running tests to detect faults related to the interface. Integration testing, in other words, is the comprehensive testing the products collection of modules.

Input

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import plotly.express as px
import seaborn as sns
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix
plt.figure(figsize=(10,6))
sns.heatmap(df.isnull())
plt.figure(figsize=(10,6))
sns.heatmap(df.corr(),annot=True)
fig, ax=plt.subplots(ncols=5, nrows=2, figsize= (20,10))
ax=ax.flatten()
index=0
for col,value in df.items():
    sns.boxplot(y=col,data=df,ax=ax[index])
index +=1
df["ph"] = df["ph"].fillna(df["ph"].mean())
```

```
df["Sulfate"] = df["Sulfate"].fillna(df["Sulfate"].mean())
df["Trihalomethanes"] =
x = df.drop("Potability",axis=1)
y = df["Potability"]
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.2)
accuracy_score_lr = accuracy_score(y_test,pred_lr)
accuracy_score_lr*100
models = pd.DataFrame({
    "Model": ["Logistic Regression",
              "Decision Tree",
              "Random Forest",
              "KNN"] ,
    "Accuracy Score" : [accuracy_score_lr,accuracy_score_dt,accuracy_score_rf,
                        accuracy_score_knn]
})
```

Test result

For the described project, a sequential analysis of integration testing was undertaken. The findings of the integration testing were positive and encouraging.

5.2.3 System Testing

System testing is a sort of software testing that is carried on a whole integrated system in order to assess the system's compliance with the requirements. Passed components are used as input in system testing. Integration testing is used to find any discrepancies between the unit that are linked together. System testing looks for flaws in both the individual components and the entire system. The behaviour of a component or a system when it is tested is the result of system testing.

Input

```
from sklearn import metrices
# Initialize the Logistic Regression model
model_lr = LogisticRegression()

# Training the model
model_lr.fit(x_train, y_train)

# Make predictions on the test set
pred_lr = model_lr.predict(x_test)

```

```
# Calculate accuracy score
accuracy_score_lr = accuracy_score(y_test, pred_lr)
accuracy_score_lr *= 100  # Multiply by 100 to get percentage

model_rf = RandomForestClassifier()
#Training Model RF
model_rf.fit(x_train,y_train)
#Making Prediction
pred_rf = model_rf.predict(x_test)
accuracy_score_rf = accuracy_score(y_test,pred_rf)
accuracy_score_rf*100
cm3 = confusion_matrix(y_test,pred_rf)
cm3
from sklearn.tree import DecisionTreeClassifier

# creating the model object
model_dt = DecisionTreeClassifier(max_depth = 4)
#Training of decision tree
model_dt.fit(x_train,y_train)
#Making prediction using Decision Tree
pred_dt = model_dt.predict(x_test)
accuracy_score_dt = accuracy_score(y_test,pred_dt)
accuracy_score_dt*100
cm2 = confusion_matrix(y_test,pred_dt)
cm2
df["ph"] = df["ph"].fillna(df["ph"].mean())
df["Sulfate"] = df["Sulfate"].fillna(df["Sulfate"].mean())
df["Trihalomethanes"] =
x = df.drop("Potability",axis=1)
y = df["Potability"]
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.2)
accuracy_score_lr = accuracy_score(y_test,pred_lr)
accuracy_score_lr*100
models = pd.DataFrame({
    "Model": ["Logistic Regression",
              "Decision Tree",
              "Random Forest",
              "KNN"] ,

    "Accuracy Score" : [accuracy_score_lr,accuracy_score_dt,accuracy_score_rf,
                        accuracy_score_knn]
})
```

5.2.4 Test Result

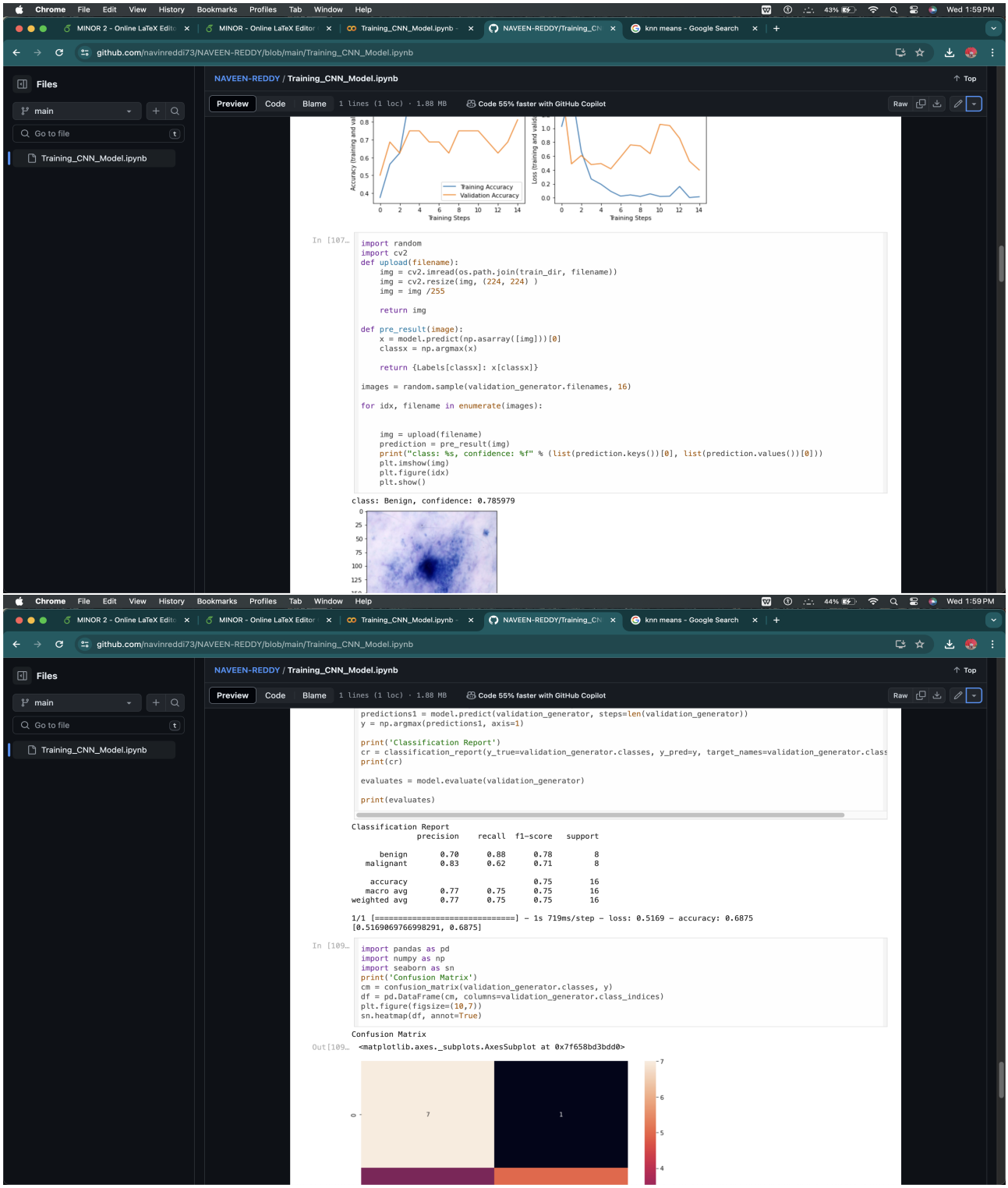


Figure 5.2: **Test Image**

The Figure 5.3 depicts the Test Image. In this project the dataset con- sists of 1500 records in the form of excel sheet in csv format as input to train algorithm.

Chapter 6

RESULTS AND DISCUSSIONS

6.1 Efficiency of the Proposed System

The proposed system is based on the Random Forest algorithm that creates many decision trees. Accuracy of proposed system is done by using random forest gives the ouput approximately 76% to 78%. Random Forest implements many decision trees and also gives the most accurate output when compared to the decision tree. Random Forest algorithm is used in the two phases.

Firstly, the RF algorithm extracts subsamples from the original samples by using the bootstrap resampling method and creates the decision trees for each testing sample and then the algorithm classifies the decision trees and implements a vote with the help of the largest vote of the classification as a final result of the classification.

The Random Forest algorithm always includes some of the steps as follows selecting the training dataset using the bootstrap random sampling method we can derive the training sets from the original dataset properties using the size of all training set the same as that of original training dataset. Building the Random forest algorithm creating a classification regression tree each of the bootstrap training set will generate the decision trees to form a random forest model, uses the trees that are not pruned. Looking at the growth of the tree, this approach is not chosen the best feature as the internal nodes for the branches but rather the branching process is a random selection of all the trees gives the best features.

6.2 Comparison of Existing and Proposed System

6.2.1 Existing System

In the existing system, a decision tree algorithm is implemented to predict whether to grant a loan or not. When using a decision tree model, the training dataset accuracy keeps improving with splits. However, it is easy to overfit the dataset, and this overfitting can go unnoticed unless cross-validation is used.

The advantages of the decision tree model are that it is very easy to interpret, and we can clearly understand the variables and the values used to split the data. Despite these advantages, the accuracy of the decision tree in the existing system is lower compared to the proposed system.

6.2.2 Proposed System

Random Forest algorithm generates more trees when compared to the decision tree and other algorithms. Specify the number of trees in the forest and specify maximum of features to be used in the each of the tree. But, we cannot control the randomness of the forest in which the feature is a part of the algorithm. Accuracy keeps increasing as we increase the number of trees but it becomes static at one certain point. Unlike the decision tree it won't create more biased and decreases variance. Proposed system is implemented using the Random forest algorithm so that the accuracy is more when compared to the existing system.

6.3 Sample Code

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import plotly.express as px
import seaborn as sns
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split

from sklearn.metrics import accuracy_score, classification_report, confusion_matrix
df = pd.read_csv("/content/drive/MyDrive/Colab Notebooks/Minor 1/water_potability.csv")
df.head()
plt.figure(figsize=(10,6))
sns.heatmap(df.isnull())
fig, ax=plt.subplots(ncols=5, nrows=2, figsize= (20,10))
ax=ax.flatten()
index=0
for col,value in df.items():
  sns.boxplot(y=col,data=df,ax=ax[index])

  index +=1

fig = px.pie(df,names = "Potability",hole = 0.4,template ="plotly_dark")
fig.show()
df["ph"] = df["ph"].fillna(df["ph"].mean())
df["Sulfate"] = df["Sulfate"].fillna(df["Sulfate"].mean())
```

```
df["Trihalomethanes"] = df["Trihalomethanes"].fillna(df["Trihalomethanes"].mean())

#Logistic Regression
from sklearn.linear_model import LogisticRegression

#object of LR
model_lr = LogisticRegression()
#Training Model
model_lr.fit(x_train,y_train)
#Making Prediction
pred_lr = model_lr.predict(x_test)
# accuracy score
accuracy_score_lr = accuracy_score(y_test,pred_lr)
accuracy_score_lr*100

from sklearn.tree import DecisionTreeClassifier

# creating the model object
model_dt = DecisionTreeClassifier(max_depth = 4)
#Training of decision tree
model_dt.fit(x_train,y_train)
#Making prediction using Decision Tree
pred_dt = model_dt.predict(x_test)
accuracy_score_dt = accuracy_score(y_test,pred_dt)
accuracy_score_dt*100
#confusion matrix
cm2 = confusion_matrix(y_test,pred_dt)
cm2


from sklearn.ensemble import RandomForestClassifier

# Creating model object
model_rf = RandomForestClassifier()
#Training Model RF
model_rf.fit(x_train,y_train)
#Making Prediction
pred_rf = model_rf.predict(x_test)
accuracy_score_rf = accuracy_score(y_test,pred_rf)
accuracy_score_rf*100
cm3 = confusion_matrix(y_test,pred_rf)
cm3


from sklearn.neighbors import KNeighborsClassifier

#Creating Model object
model_knn = KNeighborsClassifier()
for i in range(4,15):
  model_knn = KNeighborsClassifier(n_neighbors=i)
```

```
    model_knn.fit(x_train,y_train)
    pred_knn = model_knn.predict(x_test)
    accuracy_score_knn = accuracy_score(y_test,pred_knn)
    print(i,accuracy_score_knn)
model_knn = KNeighborsClassifier(n_neighbors=11)
model_knn.fit(x_train,y_train)
pred_knn = model_knn.predict(x_test)
accuracy_score_knn = accuracy_score(y_test,pred_knn)
print(accuracy_score_knn*100)

#outputs
models = pd.DataFrame({
    "Model": ["Logistic Regression",
              "Decision Tree",
              "Random Forest",
              "KNN"] ,

    "Accuracy Score" : [accuracy_score_lr,accuracy_score_dt,accuracy_score_rf,
                        accuracy_score_knn]
})
models

#accuracy
sns.barplot(x="Accuracy Score",y= "Model",data=models)
models.sort_values(by="Accuracy Score",ascending= False)
```

Output

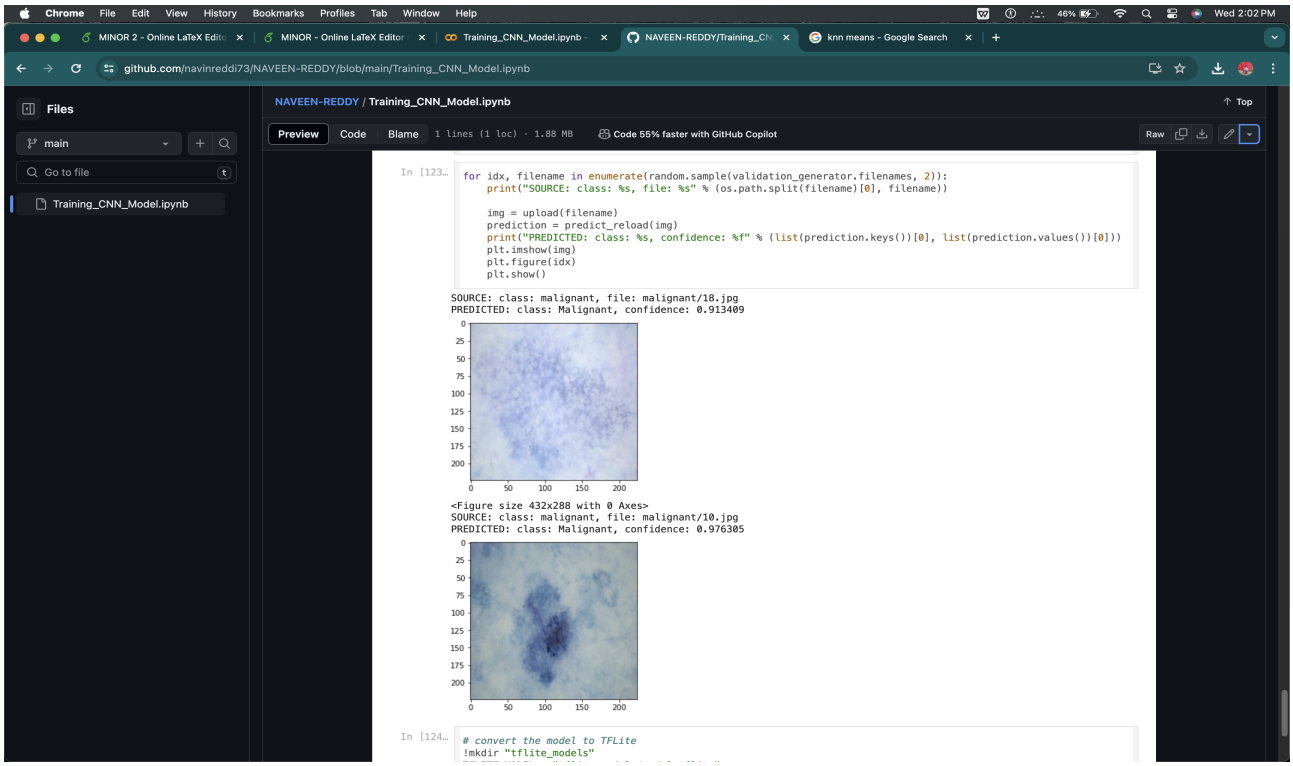


Figure 6.1: **Processed Skin Tissue**

The Figure 6.1 represents the processed skin tissue. The term "processed skin tissue" generally refers to skin samples that have undergone some form of treatment, preparation, or modification for various purposes. These processed tissues are often used in medical, research, or therapeutic contexts. Here are a few scenarios where processed skin tissue might be involved.In dermatology and plastic surgery, processed skin tissue may be used in grafting procedures. This involves taking skin from one area of the body (donor site) and placing it on another area (recipient site) that requires skin replacement, such as in the case of burns or wounds.Skin tissue samples can be processed for research purposes. This may involve fixation, embedding, and sectioning for histological analysis. Researchers may study processed skin tissue to understand skin structure, diseases, or the effects of certain treatments.Skin tissue may be processed for cryopreservation, a technique that involves freezing the tissue at extremely low temperatures to preserve it for future use.

Chapter 7

CONCLUSION AND FUTURE ENHANCEMENTS

7.1 Conclusion

The integration of machine learning in skin cancer detection represents a transformative leap forward in the realm of dermatological diagnostics. The development and implementation of advanced algorithms, particularly CNN, have showcased remarkable capabilities in accurately identifying and classifying skin lesions. The economic feasibility of such systems is underscored by their potential to revolutionize patient outcomes and healthcare practices.

The initial investments in infrastructure, data acquisition, and algorithm development are counterbalanced by the long-term benefits, including enhanced diagnostic accuracy, reduced treatment costs through early detection, and improved overall efficiency in clinical workflows.

The interpretability features integrated into these systems foster trust among healthcare professionals, a critical element in the adoption of cutting-edge technologies in clinical settings. As the field continues to evolve, embracing continuous monitoring, updates, and scalability, the economic feasibility of skin cancer detection using machine learning becomes increasingly evident, paving the way for a future where technology plays a pivotal role in advancing dermatological care.

The incorporation of interpretability features enhances transparency, addressing concerns in the medical community regarding the "black-box" nature of machine learning models. As research and development continue to progress, the collaborative efforts of technologists, healthcare professionals, and researchers pave the way for a future where machine learning significantly contributes to the early detection and treatment of skin cancer, ultimately improving patient care and outcomes in dermatology.

7.2 Future Enhancements

The future enhancements of skin cancer detection using machine learning hold tremendous promise for revolutionizing dermatological diagnostics. Anticipated developments in this domain span across various facets, with a focus on improving accuracy, interpretability, and overall clinical applicability. Advanced multi-modal integration, combining various imaging techniques, is poised to provide a more holistic understanding of skin lesions.

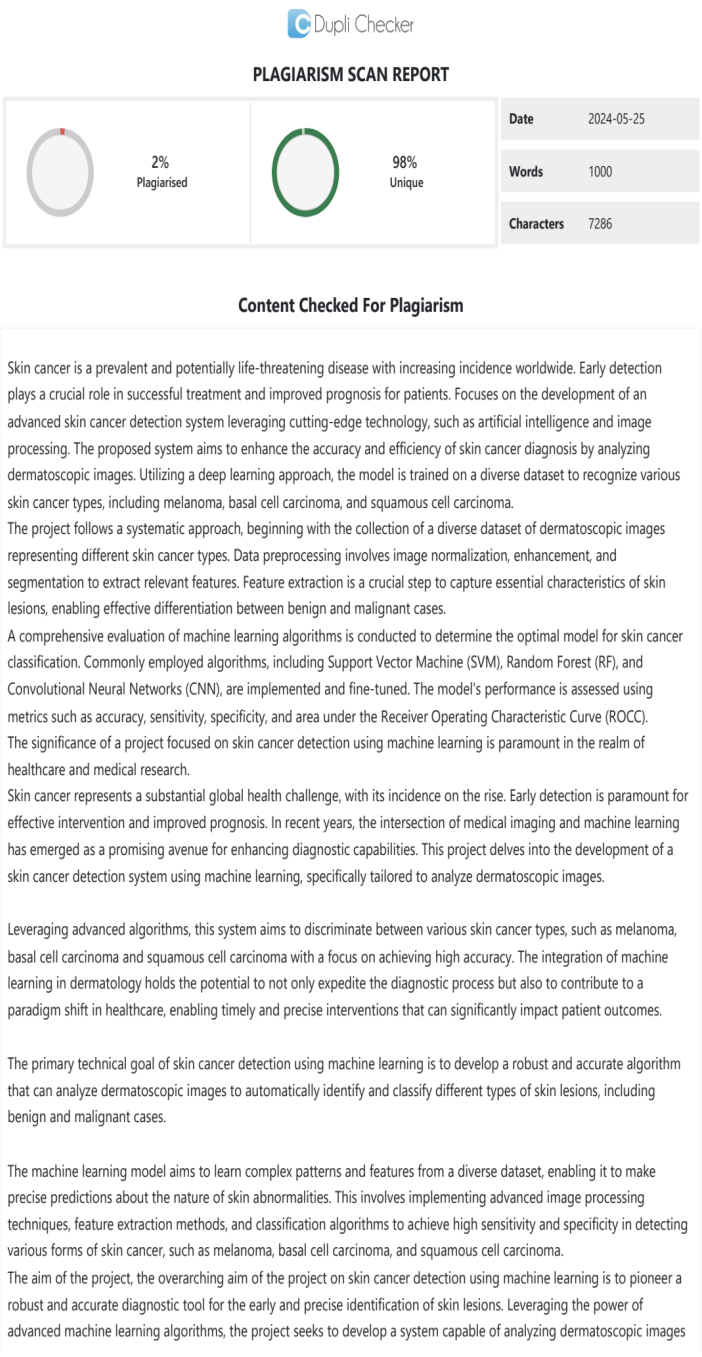
Explainable AI will take center stage, enhancing transparency in machine learning models and fostering trust among healthcare professionals. Real-time monitoring and feedback mechanisms are expected to become integral, allowing for dynamic adaptations to evolving skin conditions. Automated lesion segmentation techniques are likely to see refinement, contributing to more precise boundary delineation.

Innovations in data augmentation and synthesis will address challenges related to dataset diversity. Personalized risk stratification, considering individual patient characteristics, may become a standard feature, tailoring diagnostic recommendations for each patient. Edge computing technologies will bring skin cancer detection closer to the point of care, facilitating quicker diagnoses in diverse settings. Ethical considerations and regulatory compliance will be paramount, ensuring responsible deployment of these technologies.

Collaborative learning approaches, such as federated models, will promote knowledge sharing without compromising patient privacy. Furthermore, the integration of skin cancer detection into telemedicine platforms will enhance accessibility, allowing for remote assessments and reaching underserved populations. Collectively, these anticipated advancements signify a future where machine learning-driven skin cancer detection stands at the forefront of precision medicine, offering enhanced diagnostic capabilities with a focus on patient-centric care.

Chapter 8

PLAGIARISM REPORT



Dupli Checker

PLAGIARISM SCAN REPORT

2% Plagiarised

98% Unique

Date	2024-05-25
Words	1000
Characters	7286

Content Checked For Plagiarism

Skin cancer is a prevalent and potentially life-threatening disease with increasing incidence worldwide. Early detection plays a crucial role in successful treatment and improved prognosis for patients. Focuses on the development of an advanced skin cancer detection system leveraging cutting-edge technology, such as artificial intelligence and image processing. The proposed system aims to enhance the accuracy and efficiency of skin cancer diagnosis by analyzing dermatoscopic images. Utilizing a deep learning approach, the model is trained on a diverse dataset to recognize various skin cancer types, including melanoma, basal cell carcinoma, and squamous cell carcinoma.
The project follows a systematic approach, beginning with the collection of a diverse dataset of dermatoscopic images representing different skin cancer types. Data preprocessing involves image normalization, enhancement, and segmentation to extract relevant features. Feature extraction is a crucial step to capture essential characteristics of skin lesions, enabling effective differentiation between benign and malignant cases.
A comprehensive evaluation of machine learning algorithms is conducted to determine the optimal model for skin cancer classification. Commonly employed algorithms, including Support Vector Machine (SVM), Random Forest (RF), and Convolutional Neural Networks (CNN), are implemented and fine-tuned. The model's performance is assessed using metrics such as accuracy, sensitivity, specificity, and area under the Receiver Operating Characteristic Curve (ROCC).
The significance of a project focused on skin cancer detection using machine learning is paramount in the realm of healthcare and medical research.
Skin cancer represents a substantial global health challenge, with its incidence on the rise. Early detection is paramount for effective intervention and improved prognosis. In recent years, the intersection of medical imaging and machine learning has emerged as a promising avenue for enhancing diagnostic capabilities. This project delves into the development of a skin cancer detection system using machine learning, specifically tailored to analyze dermatoscopic images.

Leveraging advanced algorithms, this system aims to discriminate between various skin cancer types, such as melanoma, basal cell carcinoma and squamous cell carcinoma with a focus on achieving high accuracy. The integration of machine learning in dermatology holds the potential to not only expedite the diagnostic process but also to contribute to a paradigm shift in healthcare, enabling timely and precise interventions that can significantly impact patient outcomes.

The primary technical goal of skin cancer detection using machine learning is to develop a robust and accurate algorithm that can analyze dermatoscopic images to automatically identify and classify different types of skin lesions, including benign and malignant cases.

The machine learning model aims to learn complex patterns and features from a diverse dataset, enabling it to make precise predictions about the nature of skin abnormalities. This involves implementing advanced image processing techniques, feature extraction methods, and classification algorithms to achieve high sensitivity and specificity in detecting various forms of skin cancer, such as melanoma, basal cell carcinoma, and squamous cell carcinoma.
The aim of the project, the overarching aim of the project on skin cancer detection using machine learning is to pioneer a robust and accurate diagnostic tool for the early and precise identification of skin lesions. Leveraging the power of advanced machine learning algorithms, the project seeks to develop a system capable of analyzing dermatoscopic images

Figure 8.1: **Plagarism Report**

Chapter 9

SOURCE CODE & POSTER PRESENTATION

9.1 Source Code

```

import pandas as pd
import numpy as np
import seaborn as sns
import plotly.express as px
import matplotlib.pyplot as plt

from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split

from sklearn.metrics import accuracy_score, classification_report, confusion_matrix
df = pd.read_csv("/content/drive/MyDrive/Colab Notebooks/Minor 1/water_potability.csv")
df.head()
df.isnull().sum()
plt.figure(figsize=(12,8))
sns.heatmap(df.isnull())
plt.figure(figsize= (12,8))
sns.heatmap(df.corr(),annot=True)

df["Potability"].value_counts()
#Visulaization dataset also checking for outliers

fig, ax = plt.subplots(ncols=5, nrows=2, figsize = (20,10))

ax = ax.flatten()

index = 0

for col,values in df.items():
  sns.boxplot(y=col,data=df,ax=ax[index])

  index +=1
sns.pairplot(df)
fig = px.pie(df,names = "Potability",hole = 0.4,template ="plotly_dark")
fig.show()
```

```
fig = px.scatter(df,x = "ph",y="Sulfate",color = "Potability",template= "plotly_dark")
fig.show()
fig = px.scatter(df,x = "Organic_carbon",y="Hardness",color = "Potability",template= "plotly_dark")
fig.show()
df.isnull().mean().plot.bar(figsize = (10,6))
plt.xlabel("Features")
plt.ylabel("Percentage of missing values")
df["ph"] = df["ph"].fillna(df["ph"].mean())
df["Sulfate"] = df["Sulfate"].fillna(df["Sulfate"].mean())
df["Trihalomethanes"] = df["Trihalomethanes"].fillna(df["Trihalomethanes"].mean())
df.isnull().sum()
sns.heatmap(df.isnull())
df.head()
x = df.drop("Potability",axis=1)
y = df["Potability"]
x.shape , y.shape
scaler = StandardScaler()
x = scaler.fit_transform(x)
x
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.2)
x_train.shape , x_test.shape


#Logistic Regression
from sklearn.linear_model import LogisticRegression

#object of LR
model_lr = LogisticRegression()
#Training Model
model_lr.fit(x_train,y_train)
#Making Prediction
pred_lr = model_lr.predict(x_test)
# accuracy score
accuracy_score_lr = accuracy_score(y_test,pred_lr)
accuracy_score_lr*100

from sklearn.tree import DecisionTreeClassifier

# creating the model object
model_dt = DecisionTreeClassifier(max_depth = 4)
#Training of decision tree
model_dt.fit(x_train,y_train)
#Making prediction using Decision Tree
pred_dt = model_dt.predict(x_test)
accuracy_score_dt = accuracy_score(y_test,pred_dt)
accuracy_score_dt*100

#confusion matrix
cm2 = confusion_matrix(y_test,pred_dt)
cm2
```

```
86
87 from sklearn.ensemble import RandomForestClassifier
88
89 # Creating model object
90 model_rf = RandomForestClassifier()
91 #Training Model RF
92 model_rf.fit(x_train,y_train)
93 #Making Prediction
94 pred_rf = model_rf.predict(x_test)
95 accuracy_score_rf = accuracy_score(y_test,pred_rf)
96 accuracy_score_rf*100
97 cm3 = confusion_matrix(y_test,pred_rf)
98 cm3
99
100 from sklearn.neighbors import KNeighborsClassifier
101
102 #Creating Model object
103 #model_knn = KNeighborsClassifier()
104 for i in range(4,15):
105   model_knn = KNeighborsClassifier(n_neighbors=i)
106   model_knn.fit(x_train,y_train)
107   pred_knn = model_knn.predict(x_test)
108   accuracy_score_knn = accuracy_score(y_test,pred_knn)
109   print(i,accuracy_score_knn)
110 model_knn = KNeighborsClassifier(n_neighbors=11)
111 model_knn.fit(x_train,y_train)
112 pred_knn = model_knn.predict(x_test)
113 accuracy_score_knn = accuracy_score(y_test,pred_knn)
114 print(accuracy_score_knn*100)
115
116 from sklearn.svm import SVC
117
118 #Creating object of Model
119 model_svm = SVC(kernel="rbf")
120 #Model training
121 model_svm.fit(x_train,y_train)
122 #Make prediction
123 pred_svm = model_svm.predict(x_test)
124 accuracy_score_svm = accuracy_score(y_test,pred_svm)
125 accuracy_score_svm*100
126
127 from pandas.core.arrays.interval import le
128 from sklearn.ensemble import AdaBoostClassifier
129
130 #Making object of Model
131 model_ada = AdaBoostClassifier(n_estimators=200,learning_rate=0.03)
132 #Training the model
133 model_ada.fit(x_train,y_train)
134 #Making prediction
135 pred_ada = model_ada.predict(x_test)
```

```
#accuracy check
accuracy_score_ada = accuracy_score(y_test,pred_ada)
accuracy_score_ada*100

from xgboost import XGBClassifier

#create model
model_xgb = XGBClassifier(n_estimators=100,learning_rate=0.04)
#Traning Model
model_xgb.fit(x_train,y_train)
#Prediction
pred_xgb = model_xgb.predict(x_test)
#accuracy
accuracy_score_xgb = accuracy_score(y_test,pred_xgb)
accuracy_score_xgb*100
models = pd.DataFrame({
    "Model": ["Logistic Regression",
              "Decision Tree",
              "Random Forest",
              "KNN",
              "SVM",
              "AdaBoost",
              "XGBoosT"] ,

    "Accuracy Score" : [accuracy_score_lr,accuracy_score_dt,accuracy_score_rf,
                        accuracy_score_knn,accuracy_score_svm,accuracy_score_ada,accuracy_score_xgb]
})
models
sns.barplot(x="Accuracy Score",y= "Model",data=models)
models.sort_values(by="Accuracy Score",ascending= False)
```

9.2 Poster Presentation

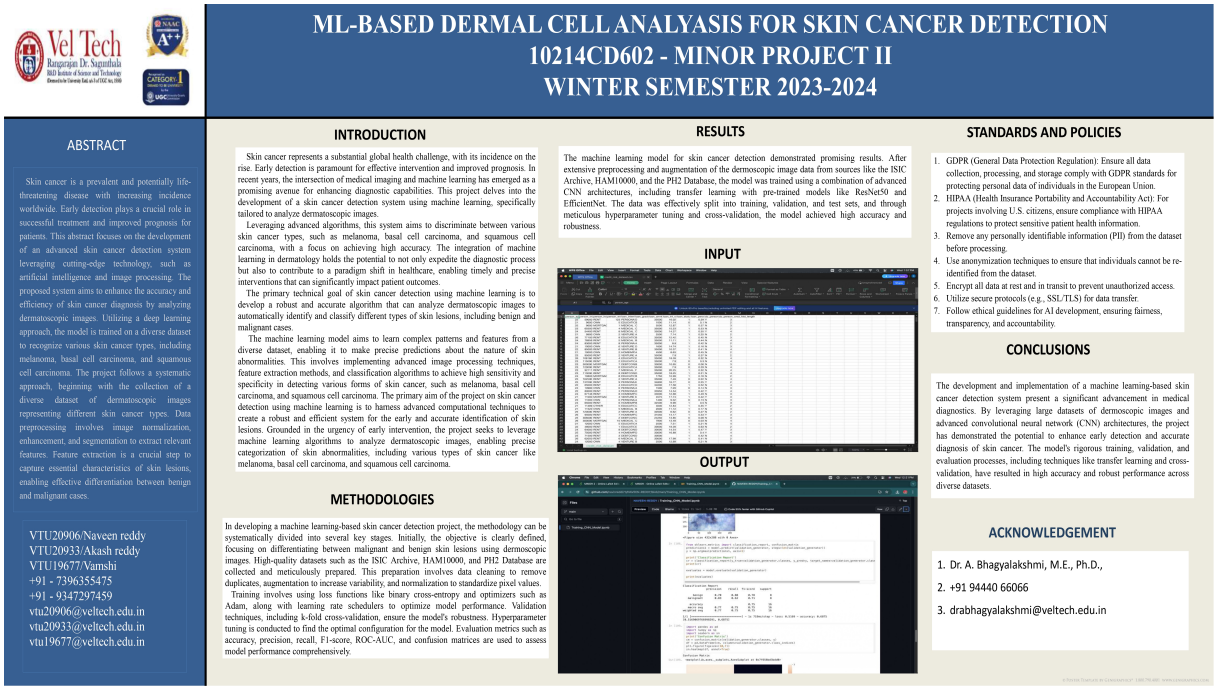


Figure 9.1: **Poster Diagram**

References

[1] A. Imran, A. Nasir, M. Bilal, G. Sun, A. Alzahrani and A. Almuhaimeed, "Skin Cancer Detection Using Combined Decision of Deep Learners" in IEEE Access, vol. 10, pp. 118198-118212, 2022.

[2] A. R. Chishti et al., "Advances in Antenna-Based Techniques for Detection and Monitoring of Critical Chronic Diseases: A Comprehensive Review" in IEEE Access, vol. 11, pp. 104463-104484, 2023.

[3] M. N. Hamza et al., "Designing a High-Sensitivity Microscale Triple-Band Biosensor Based on Terahertz MTMs to Provide a Perfect Absorber for Non-Melanoma Skin Cancer Diagnostic" in IEEE Photonics Journal, vol. 16, no. 2, pp. 1-13, April 2024.

[4] K. Mridha, M. M. Uddin, J. Shin, S. Khadka and M. F. Mridha, "An Interpretable Skin Cancer Classification Using Optimized Convolutional Neural Network for a Smart Healthcare System" in IEEE Access, vol. 11, pp. 41003-41018, 2023.

[5] R. Ashraf et al., "Region-of-Interest Based Transfer Learning Assisted Framework for Skin Cancer Detection" in IEEE Access, vol. 8, pp. 147858-147871, 2020.

[6] R. Schiavoni, G. Maietta, E. Filieri, A. Masciullo and A. Cataldo, "Microwave Reflectometry Sensing System for Low-Cost in-vivo Skin Cancer Diagnostics" in IEEE Access, vol. 11, pp. 13918-13928, 2023.

[7] O. Abuzaghleh, B. D. Barkana and M. Faezipour, "Noninvasive Real-Time Automated Skin Lesion Analysis System for Melanoma Early Detection and Prevention" in IEEE Journal of Translational Engineering in Health and Medicine, vol. 3, pp. 1-12, 2015.

[8] N. Andreasen et al., "Skin Electrical Resistance as a Diagnostic and Therapeutic Biomarker of Breast Cancer Measuring Lymphatic Regions" in IEEE Access, vol. 9, pp. 152322-152332, 2021.

[9] L. Riaz et al., "A Comprehensive Joint Learning System to Detect Skin Cancer" in IEEE Access, vol. 11, pp. 79434-79444,2023.

[10] A. A. Adegun and S. Viriri, “FCN-Based DenseNet Framework for Automated Detection and Classification of Skin Lesions in Dermoscopy Images” in IEEE Access, vol. 8, pp. 150377-150396, 2020.