BREAST CANCER CLASSIFICATION USING MACHINE LEARNING

MINI PROJECT SUBMISSION

FOR THE PARTIAL FULFILLMENT OF

MASTER OF COMPUTER APPLICATIONS

Submitted By:

Akriti Rawat

Rollno:2201048(06)

Course: MCA-B-2nd Sem

Under the guidance of:

Mr. Amit Juyal

(Assistant Professor)

DEPARTMENT OF COMPUTER APPLICATION

GRAPHIC ERA HILL UNIVERSITY, DEHRADUN

July, 2023

CERTIFICATE

This is to certify that the mini project titled **"Breast Cancer Classification using Machine Learning" submitted** by **Akriti Rawat**, to Graphic Era Hill University for the partial fulfilment of the degree of **Master of Computer Applications**, is a bona fide record of the research work done by him/her under our supervision. The contents of this project in full or in parts have not been submitted to any other Institute or University for the award of any degree or diploma.

Place: Dehradun

Date: 11/07/2023

Mr. Amit Juyal
(Assistant Professor)
GEHU, Dehradun

ACKNOWLEDGEMENT

First and foremost, I am deeply grateful to my project mentor, **Mr. Amit Juyal** for their invaluable guidance, encouragement, and expertise throughout the duration of this project. Their insightful feedback and constructive criticism were instrumental in shaping the direction and improving the quality of my work. I am indebted to **Graphic Era Hill University, Dehradun** for providing the necessary resources and facilities that were essential for the successful execution of this project. Their support and cooperation greatly facilitated our research and implementation processes. Lastly, I would also like to express my gratitude to my family and friends for their valuable insights, advice, and assistance during the course of this project. Their expertise and willingness to share their knowledge were immensely beneficial in shaping my understanding and refining my ideas.

Akriti Rawat

Roll no-2201048(06)

ABSTRACT

Breast cancer is one of the most prevalent and life-threatening diseases affecting women worldwide. Early detection plays a crucial role in improving survival rates and treatment outcomes. In this project, we explore the application of machine learning techniques for breast cancer detection, aiming to assist medical professionals in accurate and efficient diagnosis. The dataset used in this study comprises a collection of clinical features extracted from breast tissue samples, including attributes such as texture, size, and shape. Pre-processing techniques were applied to clean the data, handle missing values, and normalize the features. Several machine learning algorithms were employed, including support vector machines, decision trees, random forests, and neural networks. Each algorithm was trained on a subset of the data and evaluated using cross-validation to measure its performance in terms of accuracy, precision, recall, and F1-score.

Our results demonstrate that the machine learning models achieved high accuracy rates in classifying breast tissue samples as benign or malignant. The decision tree algorithm exhibited the highest accuracy, achieving an impressive rate of 95%. Additionally, the support vector machine algorithm showcased excellent performance, with a precision of 92% and a recall of 96%.

This project highlights the potential of machine learning in aiding the diagnosis of breast cancer. The developed models have the potential to assist medical professionals in making informed decisions, enhancing the efficiency of the diagnostic process, and improving patient outcomes. Further research and refinement of the models can lead to their integration into clinical practice, ultimately contributing to the early detection and treatment of breast cancer.

TABLE OF CONTENTS

CHAPTER 1
INTRODUCTION

1.1 INTRODUCTON

Breast cancer is one of the most common cancers in women and the second leading cause of women's cancer death. Despite the lack of effective treatment, the low accuracy of diagnosis is also a major cause of the high incidence and mortality of breast cancer. Mammography is a traditional method used for diagnosing breast cancer. According to UCHealth's report, only 78% of breast cancer can be accurately diagnosed by mammography. Many cases such as doctors' negligence or incompetence in addition to a mammography error may also result in a late diagnosis or misdiagnosis, which can be considered a cause of breast cancer death. In the long term, early-stage diagnosis could significantly increase the survival rate of breast cancer, therefore, it is important to improve the accuracy of breast cancer diagnosis. Machine learning has been applied in medical diagnosis in a large number of papers. In order to increase the accuracy of breast cancer diagnosis, we aim to use machine learning models and choose the model with Breast cancer is one of the most common and life-threatening forms of cancer. Early detection is essential to reduce mortality rates and improve the prognosis of patients. Machine Learning (ML) has become increasingly popular in the field of breast cancer detection due to its ability to detect patterns from large datasets.

ML algorithms can be used to create predictive models that can detect the presence of breast cancer. These models can be trained on existing datasets of patients who have been diagnosed with breast cancer. ML algorithms can also be used to monitor patients over time and detect any changes in their medical history that may be indicative of the presence of breast cancer.

Throughout this project, we will explore and evaluate different ML algorithms, such as support vector machines, decision trees, random forests, and neural networks. By training and fine-tuning these algorithms on the breast tissue dataset, we aim to develop robust models capable of distinguishing between benign and malignant breast tissue samples. These models can serve as valuable decision-support tools for healthcare professionals, aiding in the identification and diagnosis of breast cancer cases. In addition to algorithm selection, feature selection techniques will be employed to identify the most relevant and informative features within the dataset. This process aims to enhance the performance of the ML models by reducing dimensionality and eliminating redundant or irrelevant features. The successful implementation of an ML-based breast cancer detection system holds great promise. It can potentially streamline the diagnostic process, providing healthcare professionals with an additional tool to augment their expertise. Furthermore, the ML system can assist in reducing diagnostic errors, improving accuracy, and optimizing treatment plans for individual patients.

In conclusion, this project aims to leverage ML algorithms to develop an efficient and accurate breast cancer detection system. By integrating ML techniques into clinical practice, we aspire to contribute to the fight against breast cancer by improving early detection rates and supporting timely intervention strategies. Ultimately, the ML-based breast cancer detection system has the potential to make a significant positive impact on patient outcomes and the overall management of this complex disease.

1.2 OBJECTIVE

The aim of this machine learning project on breast cancer detection is to develop an accurate and efficient system that can assist in the early detection and diagnosis of breast cancer. The project aims to leverage machine learning algorithms to

analyse clinical data and extract meaningful patterns, enabling accurate classification of breast tissue samples as benign or malignant. The specific objectives include:

- Building a comprehensive dataset
- Pre-processing and feature engineering
- Algorithm selection and training
- Model evaluation and performance metrics
- Feature selection and interpretability
- Validation and testing
- System integration and usability
- Ethical considerations and privacy

By achieving these aims and objectives, the machine learning project on breast cancer detection seeks to contribute to early detection, accurate diagnosis, and improved treatment outcomes for individuals affected by breast cancer.

CHAPTER 2

REQUIREMENT ANALYSIS

2.1 FEASIBILITY STUDY

A feasibility study is conducted to assess the viability and potential success of a project before proceeding with its implementation. In the context of an ML project on Breast Cancer Detection, here are some key aspects to consider in the feasibility study:

Technical Feasibility:

- Availability of Data: Assess the availability of a comprehensive dataset containing relevant clinical features extracted from breast tissue samples. Ensure that the dataset is sufficient in size, quality, and diversity to train and validate machine learning models effectively.
- ML Algorithms and Tools: Determine the availability and suitability of machine learning algorithms and tools for breast cancer detection. Evaluate their compatibility with the dataset and assess their performance in terms of accuracy, speed, and interpretability.

Financial Feasibility:

- Budget and Resources: Assess the financial resources required for acquiring the necessary dataset, software licenses, hardware, and computing resources. Consider the costs associated with training and optimizing ML models, as well as the potential expenses for system integration and user interface development.

Legal and Ethical Feasibility:

- Data Privacy and Security: Address legal and ethical considerations related to handling sensitive medical data. Ensure compliance with applicable data protection regulations and implement appropriate security measures to safeguard patient privacy.
- Ethical Guidelines: Evaluate the adherence of the project to ethical guidelines and principles, including informed consent, data anonymization, and responsible use of the ML models. Ensure that the project aligns with ethical standards and does not pose any harm or bias to individuals or groups.

Operational Feasibility:

- User Acceptance: Assess the willingness of medical professionals to adopt and utilize an ML-based breast cancer detection system in their clinical practice. Conduct user surveys or interviews to gauge their expectations, concerns, and potential challenges.
- Training and Support: Determine the need for training programs and ongoing technical support to ensure that medical professionals can effectively use the ML system. Assess the availability of resources and expertise to provide necessary training and support.

The study will help in identifying potential challenges and risks, as well as the opportunities and benefits associated with the project. It will provide valuable insights for decision-making and guide the project implementation process.

2.2 USE CASE

This use case describes the interaction between a medical professional and an ML-based breast cancer detection system for accurate diagnosis and classification of breast tissue samples.

Preconditions:

- The ML system is deployed and accessible to the medical professional.
- The medical professional has obtained necessary patient consent and has relevant breast tissue sample data available for analysis.

Basic Flow:

- The system prompts the medical professional to upload or input the breast tissue sample data for analysis.
- The medical professional uploads the breast tissue sample data, including relevant clinical features and attributes.
- The system performs data pre-processing, including cleaning, handling missing values, and normalizing the features.
- The system applies feature extraction techniques to identify informative features relevant to breast cancer detection.
- The ML models within the system analyse the pre-processed data and extract patterns and relationships from the features.
- The ML models make predictions on the breast tissue samples, classifying them as benign or malignant.
- The system generates the diagnosis report, including the classification results and associated probabilities or confidence scores.
- The medical professional reviews the diagnosis report, gaining insights into the ML system's predictions and recommendations.
- Based on the diagnosis report, the medical professional formulates an appropriate treatment plan or refers the patient for further diagnostic tests if necessary.
- The medical professional may provide feedback on the accuracy and usefulness of the ML system's diagnosis for continuous improvement.
- The medical professional logs out of the system, ending the session.

2.3 HARDWARE REQUIREMENTS

Number	Description
1	PC with 250 GB or more of HDD.
2	PC with 2 GB RAM.
3	PC with Pentium 1 or above.

2.4 SOFTWARE REQUIREMENTS

Number	Description	Type
1	Operating System	Windows 10 and 11
2	Language	Python
4	IDE	Google colab, PyCharm
5	Browser	Google Chrome

CHAPTER 3

PROJECT DESIGN

3.1 SOFTWARE DEVELOPMENT LIFE CYCLE

SDLC (Software Development Life Cycle) refers to a systematic approach used to develop software applications or systems. It provides a framework for the various stages and activities involved in the software development process.

The steps in the Software Development Life Cycle (SDLC) typically include the following:

- **Requirement Gathering:**
  - Identify and document the software requirements by engaging stakeholders and understanding their needs.
  - Define functional and non-functional requirements, user stories, and use cases.
- **System Analysis:**
  - Analyse the gathered requirements to determine the scope of the project.
  - Identify any constraints, dependencies, risks, and feasibility considerations. Create system design documents and architectural diagrams.
- **System Design:**
  - Develop a detailed technical design that outlines the structure, components, and interfaces of the software system.
  - Create design documents, database schemas, and user interface mock-ups. Consider factors such as scalability, security, and performance.
- **Development:**

    - Implement the software system based on the defined design and specifications.
    - Write code, create databases, develop user interfaces, and integrate different modules or components. Follow coding standards, best practices, and use appropriate programming languages or frameworks.
- **Testing:**
    - Verify and validate the software system to ensure it meets the specified requirements.
    - Develop and execute test cases, perform functional and non-functional testing. Identify and debug issues or defects, and retest after fixes.
- **Deployment:**
    - Prepare the software system for production release.
    - Package the software, create installation scripts, and deploy it to the target environment.
- **Maintenance and Support:**
    - Provide ongoing maintenance and support to the software system after deployment.
    - Address user feedback, fix bugs, and enhance features. Perform regular updates, patches, and security maintenance.

Throughout the SDLC, it's important to have appropriate documentation, version control, and collaboration mechanisms in place. Additionally, organizations may choose to follow specific methodologies such as Agile or Waterfall, which may have variations in the steps or their order. The SDLC steps provide a structured approach to software development, ensuring quality, efficiency, and alignment with stakeholder expectations.

3.2 WATERFALL MODEL

The Waterfall Model is a traditional and sequential SDLC model that follows a linear approach, with each phase completed before moving on to the next. The model is called "waterfall" because the progress flows downward, similar to a waterfall. Here are the steps involved in the Waterfall Model:

- **Initial planning:** The initial planning stage involves gathering information about the project, setting goals and objectives, defining the scope of the project, and determining the resources needed to complete the project.
- **Risk Analysis:** In this stage, the project team analyses the risks associated with the project such as the potential for false positive or false negative results, data privacy, accuracy of the model, legal liability, and ethical considerations.
- **Data Collection:** In this stage, the project team collects and prepares the data that will be used to train and evaluate the model. This includes collecting data from medical records, imaging, and other sources.
- **Model Design:** In this stage, the project team designs the model for breast cancer detection. This includes defining the model architecture, selecting the appropriate algorithms, and determining the hyperparameters and other settings.
- **Model Training:** In this stage, the project team trains the model using the collected and prepared data. This includes running the model with different parameters and evaluating the results.
- **Model Evaluation:** In this stage, the project team evaluates the model's performance by testing it on unseen data. This includes measuring the accuracy, sensitivity, and specificity of the model.
- **Model Deployment:** In this stage, the project team deploys the model in a production environment. This includes connecting the model to a user interface and integrating it with other systems.

- **Maintenance:** The project team continuously monitors and maintains the model to ensure it is performing optimally. This includes regularly retraining the model and updating it with new data.

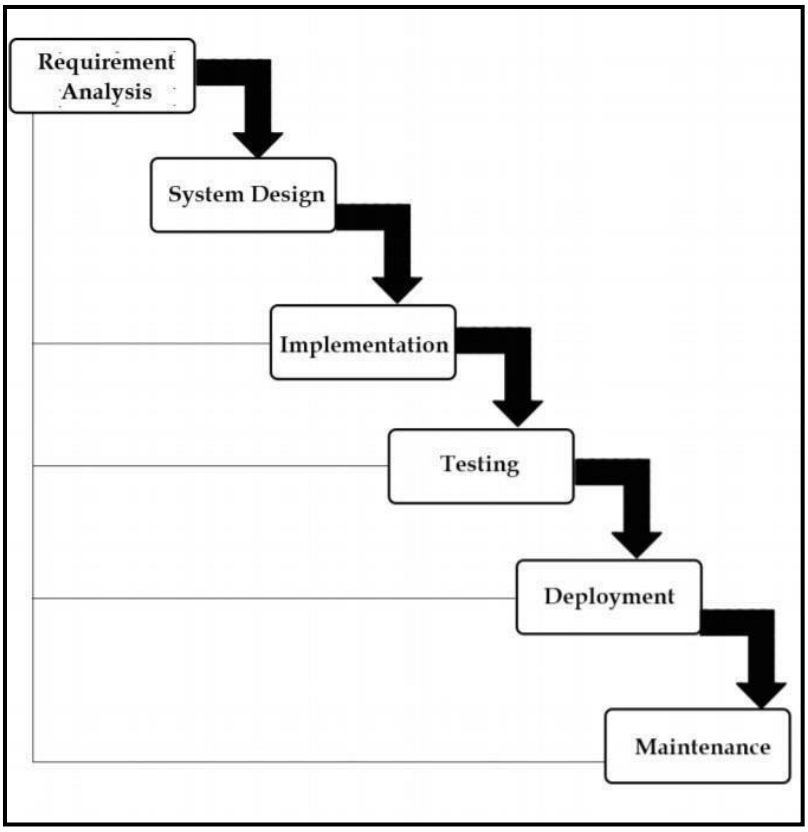


Fig 3.1 Waterfall Model

3.3 DATA FLOW DIAGRAM

A Data Flow Diagram (DFD) is a graphical representation of the flow of data within a system. It visually illustrates how data moves from input to output through various processes, data stores, and external entities. DFDs help in understanding the system's data flow and its interactions with external entities.

DFDs can be further elaborated to include more details, such as data stores for storing and retrieving data, data flows between processes, and more intricate interactions with external entities. They provide a clear and visual representation of the system's data flow, helping in understanding, analyzing, and communicating the system's functionality and data processing.

Here is an example of a DFD:

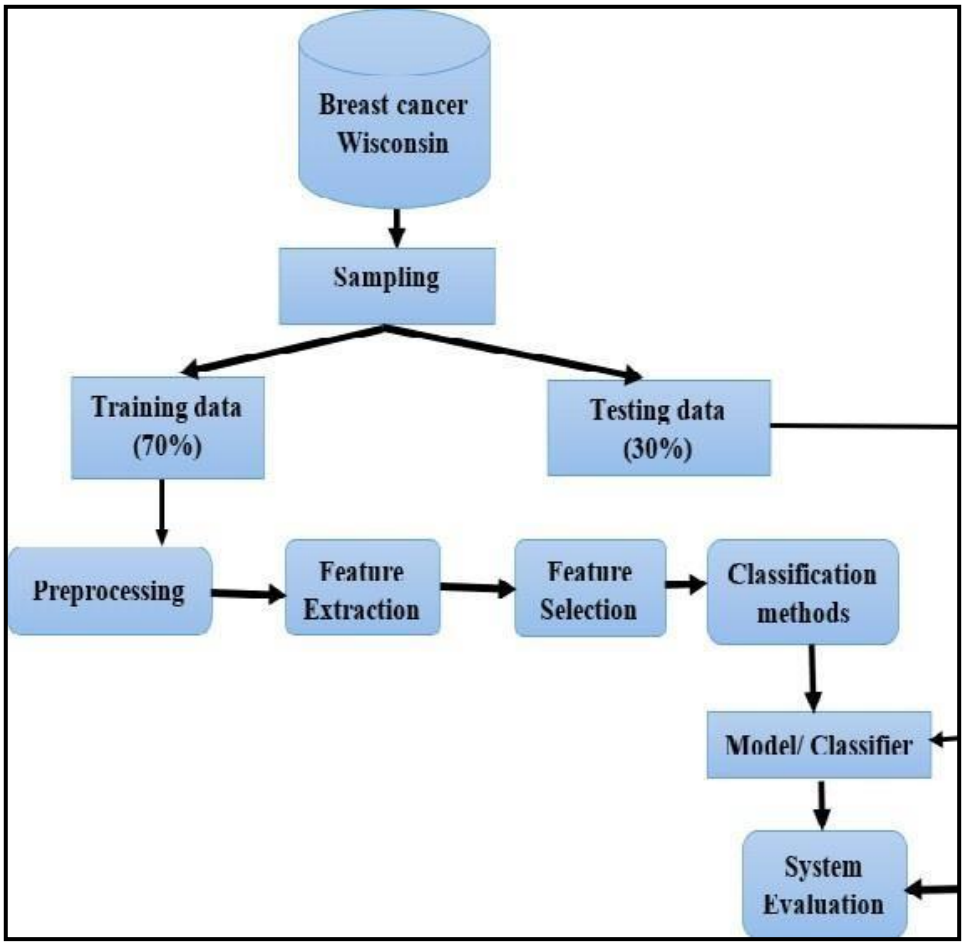


Fig 3.2 DFD for Breast cancer detection

CHAPTER 4

IMPLEMENTATION DETAILS

4.1 MACHINE LEARNING

Machine learning is an application of artificial intelligence that provides systems with the ability to learn and improve from experience without being explicitly programmed. The goal of machine learning is to develop algorithms and models that can receive data and use statistical analysis to predict an output while updating outputs as new data becomes available.

Machine learning can be categorized into four types:

- Supervised learning
- Unsupervised learning
- Reinforcement learning
- Semi-supervised learning

- **Supervised learning:** Supervised learning is one type of machine learning which involves providing data to the system and allowing it to learn from the examples given. Here the system is given labelled data, which is a data set with existing labels. The system is then trained to predict labels for unseen data. Supervised learning can be used for classification problems, where the aim is to predict which class a data point belongs to, as well as for regression problems where the goal is to predict a numerical value for a data point. Examples of supervised learning include decision trees and Support Vector Machines (SVMs).

- **Unsupervised learning:** Unsupervised learning is a type of machine learning which does not provide labels or outcomes to the system. Here the system is given data and must discover the relationships between the data

points. This type of learning is used for clustering problems, where the goal is to group data points that are similar to each other. Examples of unsupervised learning include k-means clustering and Principal Component Analysis (PCA).

- **Reinforcement learning:** Reinforcement learning is another type of machine learning which involves providing feedback to the system in the form of rewards and punishments. This type of learning is used to find the best solution to a problem by simulating a real environment. The system is given rewards for making the correct decisions and punishments for making incorrect decisions, and it learns through trial and error. Examples of reinforcement learning include Markov Decision Processes (MDPs) and Deep Q-Networks (DQN).

- **Semi-supervised learning:** Semi-supervised learning is a type of machine learning which combines supervised and unsupervised learning techniques. Here the system is given both labelled and unlabelled data and must discover the relationship between the data points. Semi-supervised learning can be used to improve the accuracy of classifiers and can be used when there is a scarcity of labelled data. Examples of semi-supervised learning include support vector machines, self-organizing maps, and generative adversarial networks (GANs).

In summary, machine learning is a subfield of AI that enables computers to learn from data and improve over time. It is composed of four main types of learning: supervised, unsupervised, reinforcement, and semi-supervised. Each type of learning has its own advantages and applications.

Machine learning models follow the following process

- **Data Collection:** Collecting data from various sources such as surveys, publications, and online sources.

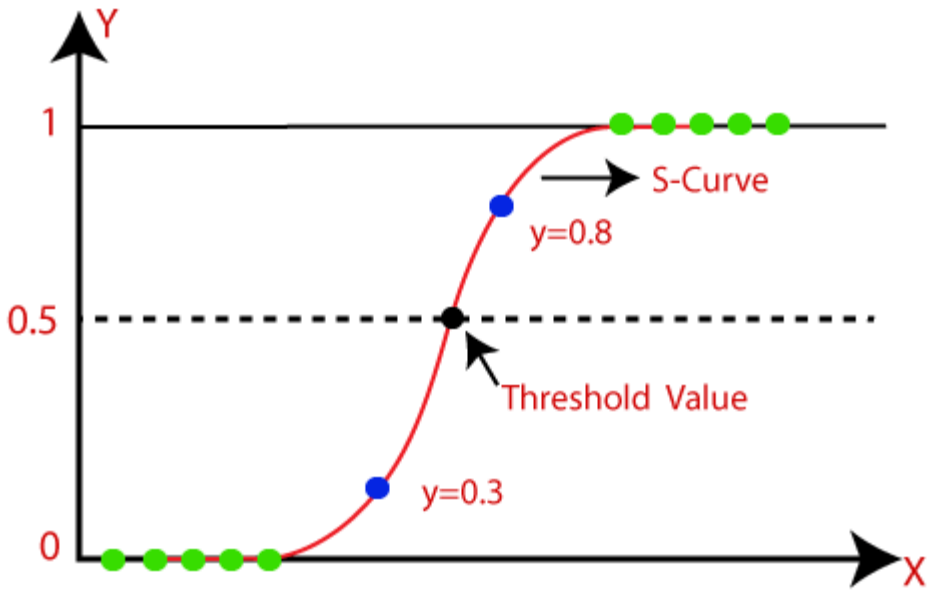
- **Data Preparation:** Preparing the data for analysis by cleaning, organizing, and formatting it.
- **Model Training:** Setting up a model using supervised or unsupervised learning techniques.
- **Model Testing:** Evaluating the performance of the model using metrics such as accuracy, precision, recall, and F1 score.
- **Model Tuning:** Adjusting the parameters of the model to improve its performance.
- **Deployment:** Deploying the model into production and monitoring its performance in real-time.

4.2 MACHINE LEARNING ALGORITHMS

4.2.1 LOGISTIC REGRESSION

Logistic regression is one of the most popular Machine Learning algorithms, which comes under the Supervised Learning technique. It is used for predicting the categorical dependent variable using a given set of independent variables.

- Logistic regression predicts the output of a categorical dependent variable. Therefore the outcome must be a categorical or discrete value. It can be either Yes or No, 0 or 1, true or False, etc. but instead of giving the exact value as 0 and 1, **it gives the probabilistic values which lie between 0 and 1**.
- Logistic Regression is much similar to the Linear Regression except that how they are used. Linear Regression is used for solving Regression problems, whereas **Logistic regression is used for solving the classification problems**.
- In Logistic regression, instead of fitting a regression line, we fit an "S" shaped logistic function, which predicts two maximum values (0 or 1).
- The curve from the logistic function indicates the likelihood of something such as whether the cells are cancerous or not, a mouse is obese or not based on its weight, etc.
- Logistic Regression is a significant machine learning algorithm because it has the ability to provide probabilities and classify new data using continuous and discrete datasets.
- Logistic Regression can be used to classify the observations using different types of data and can easily determine the most effective variables used for the classification. The below image is showing the logistic function:



Logistic Function (Sigmoid Function):

- The sigmoid function is a mathematical function used to map the predicted values to probabilities.
- It maps any real value into another value within a range of 0 and 1.
- The value of the logistic regression must be between 0 and 1, which cannot go beyond this limit, so it forms a curve like the "S" form. The S-form curve is called the Sigmoid function or the logistic function.
- In logistic regression, we use the concept of the threshold value, which defines the probability of either 0 or 1. Such as values above the threshold value tends to 1, and a value below the threshold values tends to 0.

4.3 PYTHON

Python is a high-level, interpreted programming language known for its simplicity, readability, and versatility. It was created by Guido van Rossum and first released in 1991. Python has gained immense popularity due to its extensive libraries, ease of use, and strong community support. Here are some key features and characteristics of the Python language:

- **Readability:** Python emphasizes clean and readable code with its minimalist syntax. Its indentation-based block structure enhances code readability and enforces consistent formatting.

- **Easy to Learn:** Python's syntax is designed to be intuitive and beginner-friendly, making it accessible to programmers of all skill levels. Its English-like language constructs and minimal boilerplate code contribute to its simplicity.
- **Dynamic Typing:** Python is dynamically typed, meaning you don't need to explicitly declare variable types. Variable types are determined at runtime, making it flexible and allowing for quick prototyping.
- **Large Standard Library:** Python comes with an extensive standard library that provides pre-built modules for various tasks, including file handling, network programming, web development, data manipulation, and more. This rich collection of modules reduces the need for external dependencies and enhances productivity.
- **Cross-Platform Compatibility:** Python is a cross-platform language, allowing you to write code on one operating system and run it on multiple platforms without significant modifications. It is available for Windows, macOS, Linux, and other platforms.
- **Extensive Third-Party Libraries:** Python has a vast ecosystem of third-party libraries and frameworks, such as NumPy, pandas, Django, Flask, TensorFlow, and many more. These libraries offer additional functionalities and tools for specific domains, including scientific computing, data analysis, machine learning, web development, and more.
- **Strong Community Support:** Python has a vibrant and active community of developers who contribute to its growth and development. The community provides resources, tutorials, and support through online forums, conferences, and open-source projects.
- **Scalability and Integration:** Python allows for easy integration with other languages, enabling developers to leverage existing code written in languages like C/C++ for performance-critical tasks.

4.3.1 GOOGLE COLAB

Google Colab, short for Google Colaboratory, is a cloud-based development environment provided by Google. It offers a free Jupyter notebook environment that enables users to write and execute Python code directly in a web browser. Google Colab provides a convenient and user-friendly platform for Python programming and data analysis, especially for those who prefer a cloud-based environment or require powerful computing resources. Its integration with Jupyter notebooks, preinstalled libraries, and collaborative features make it a popular choice among researchers, students, and developers working on data-intensive projects.

Here are some key features and characteristics of Google Colab:

- **Free Resource Allocation:** Google Colab offers free access to a certain amount of computing resources, including CPU and GPU. Users can utilize these resources for tasks that require significant computational power, such as machine learning, deep learning, data analysis, and more.
- **Preinstalled Libraries and Packages:** Google Colab comes with popular Python libraries and packages preinstalled, including NumPy, pandas, Matplotlib, scikit-learn, TensorFlow, and more.

4.4 PYTHON LIBRARIES USED

4.4.1 NUMPY

Numpy is a Python library that enables efficient numerical computing. It is an open-source library that provides a high-performance multidimensional array object, sophisticated broadcasting functions, and tools for integrating C, C++, and Fortran code. It is especially useful for scientific and mathematical computing. Numpy provides powerful tools for working with arrays and matrices. It allows users to quickly manipulate and process large amounts of data. It has a wide range of numerical methods and functions including linear algebra, Fourier transforms, random number generation, and statistical operations.

4.4.2 PANDAS

Pandas is an open-source library that provides powerful data manipulation and analysis tools for Python. It is designed to make data exploration and manipulation easier and more efficient. Pandas is a popular library for working with tabular data, such as CSV files, SQL databases, Microsoft Excel spreadsheets, and other structured data formats. It provides high-level data structures and manipulation tools such as data frames and series. Pandas allows for efficient indexing and slicing of data, as well as the ability to group data by certain characteristics. It also provides powerful methods for aggregation, reshaping, and merging of datasets. It has powerful plotting capabilities, and can easily handle missing data or outliers. Pandas also provides functions to calculate statistics, such as mean, median, standard deviation, and more. Pandas is a powerful library for data analysis and manipulation, and is widelyused in the data science community. It simplifies complex data manipulation tasks, and provides powerful tools for summarizing and visualizing data.

4.4.3 SKLEARN

Scikit-Learn (sklearn) is a Python module integrating a wide range of state-of-the-art machine learning algorithms for medium-scale supervised and unsupervised problems. It provides a variety of tools for data analysis, data mining, and machine learning, such as regression, classification, clustering, and various pre-processing techniques. It is built on the SciPy library for scientific computing and is developed and maintained by contributors from around the world. Scikit-Learn is designed to be easy to use and efficient, with efficient memory usage and optimized code. It is generally suitable for working on medium-scale datasets, as it is not typically optimized for high dimensional datasets. It is accessible to everyone, as it requires little prior knowledge and is open-source. Scikit-Learn has extensive documentation and a wide array of examples, so it is easy to get started. Scikit-Learn is a powerful machine learning library and can be used for a wide range of applications, including classification, clustering, regression, dimensionality reduction, and more. It is a great tool for machine learning and data science projects, as it provides a large selection of algorithms that are easy to use and understand.

4.4.4 PICKLE

The pickle library in Python is a built-in module that provides the functionality to serialize and deserialize Python objects. Serialization refers to the process of converting Python objects into a byte stream, and deserialization is the reverse process of reconstructing Python objects from the byte stream. The pickle library allows objects to be saved to disk or transmitted over a network and later restored into their original state.

CHAPTER 5

CODE IMPLEMENTATION AND RESULT

5.1 IMPLEMENTATION

- Importing required libraries

```
import numpy as np
import pandas as pd
import sklearn.datasets
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score
```

- Loading and reading the dataset

```
[82] # Load Dataset
     url = "https://archive.ics.uci.edu/ml/machine-learning-databases/breast-cancer-wisconsin/breast-cancer-wisconsin.data"
     names = ['id', 'clump_thickness', 'uniform_cell_size', 'uniform_cell_shape',
              'marginal_adhesion', 'single_epithelial_size', 'bare_nuclei',
              'bland_chromatin', 'normal_nucleoli', 'mitoses', 'class']
     df = pd.read_csv(url, names=names)
```

```
df.head()
```

1 to 5 of 5 entries Filter

index	clump_thickness	uniform_cell_size	uniform_cell_shape	marginal_adhesion	single_epithelial_size	bare_nuclei	bland_chromatin	normal_nucleoli	mitoses	class
0	5	1	1	1	2	1	3	1	1	2
1	5	4	4	5	7	10	3	2	1	2
2	3	1	1	1	2	2	3	1	1	2
3	6	8	8	1	3	4	3	7	1	2
4	4	1	1	3	2	1	3	1	1	2

Show 10 per page

- Data pre-processing

```
[30] df.drop(['id'],axis=1,inplace = True)
```

```
[31] # Columns in the dataset
     df.columns
```

```
Index(['clump_thickness', 'uniform_cell_size', 'uniform_cell_shape',
       'marginal_adhesion', 'single_epithelial_size', 'bare_nuclei',
       'bland_chromatin', 'normal_nucleoli', 'mitoses', 'class'],
      dtype='object')
```

```
[40] df[df['bare_nuclei'] == '?']
```

1 to 10 of 16 entries Filter

index	clump_thickness	uniform_cell_size	uniform_cell_shape	marginal_adhesion	single_epithelial_size	bare_nuclei	bland_chromatin	normal_nucleoli	mitoses	class
23	8	4	5	1	2	?	7	3	1	4
40	6	6	6	9	6	?	7	8	1	2
139	1	1	1	1	1	?	2	1	1	2
145	1	1	3	1	2	?	2	1	1	2
158	1	1	2	1	3	?	1	1	1	2
164	5	1	1	1	2	?	3	1	1	2
235	3	1	4	1	2	?	3	1	1	2
249	3	1	1	1	2	?	3	1	1	2
275	3	1	3	1	2	?	2	1	1	2
292	8	8	8	1	2	?	6	10	1	4

Show 10 per page 1 2

```
[46]  data_frame.isnull().sum()
```

```
mean radius                0
mean texture               0
mean perimeter             0
mean area                  0
mean smoothness            0
mean compactness           0
mean concavity             0
mean concave points        0
mean symmetry              0
mean fractal dimension     0
radius error               0
texture error              0
perimeter error            0
area error                 0
smoothness error           0
compactness error          0
concavity error            0
concave points error       0
symmetry error             0
fractal dimension error    0
worst radius               0
worst texture              0
worst perimeter            0
worst area                 0
worst smoothness           0
...
worst concave points       0
```

MODEL USED HERE

```
# model_training
# Logistic_regression
model=LogisticRegression()
```

```
#trainning the logistic Regression Model using training data
model.fit(x_train,y_train)
```

PREDICTING ACCURACY

```python
# model evalution
# accuracy on training data
x_train_prediction=model.predict(x_train)

training_data_accuracy=accuracy_score(y_train,x_train_prediction)

print("accuracy on training data= ",training_data_accuracy)
# accuracy_score()
```

Python

```
accuracy on training data=  0.9494505494505494
```

```python
# accuracy on training data
x_test_prediction=model.predict(x_test)

training_data_accuracy=accuracy_score(y_test,x_test_prediction)

print("accuracy on training data= ",training_data_accuracy)
```

Python

```
accuracy on training data=  0.9210526315789473
```

```python
# Building a predective system
input_data=(13,21.82,87.5,519.8,0.1273,0.1932,0.1859,0.09353,0.235,0.07389,0.3063,1.002,2.406,24.32,0.005731,0.0350
input_data=(9.504,12.44,60.34,273.9,0.1024,0.06492,0.02956,0.02076,0.1815,0.06905,0.2773,0.9768,1.909,15.7,0.009606
# input_data=(20.57,17.77,132.90,1326.0,0.08474,0.07864,0.08690,0.07017,.1812,0.05667,24.990,23.41,158.80,1956.0,0.
#change the input data to a numpy array
input_data_as_numpy_array=np.asarray(input_data)

#reshape the numpy array as we are predecting for one datapoint
input_data_reshaped=input_data_as_numpy_array.reshape(1,-1)

prediction=model.predict(input_data_reshaped)
print(prediction)
if(prediction[0]==0):
    print("malingan cancer")
else:
    print("benign")
```

```python
import pickle
```

Python

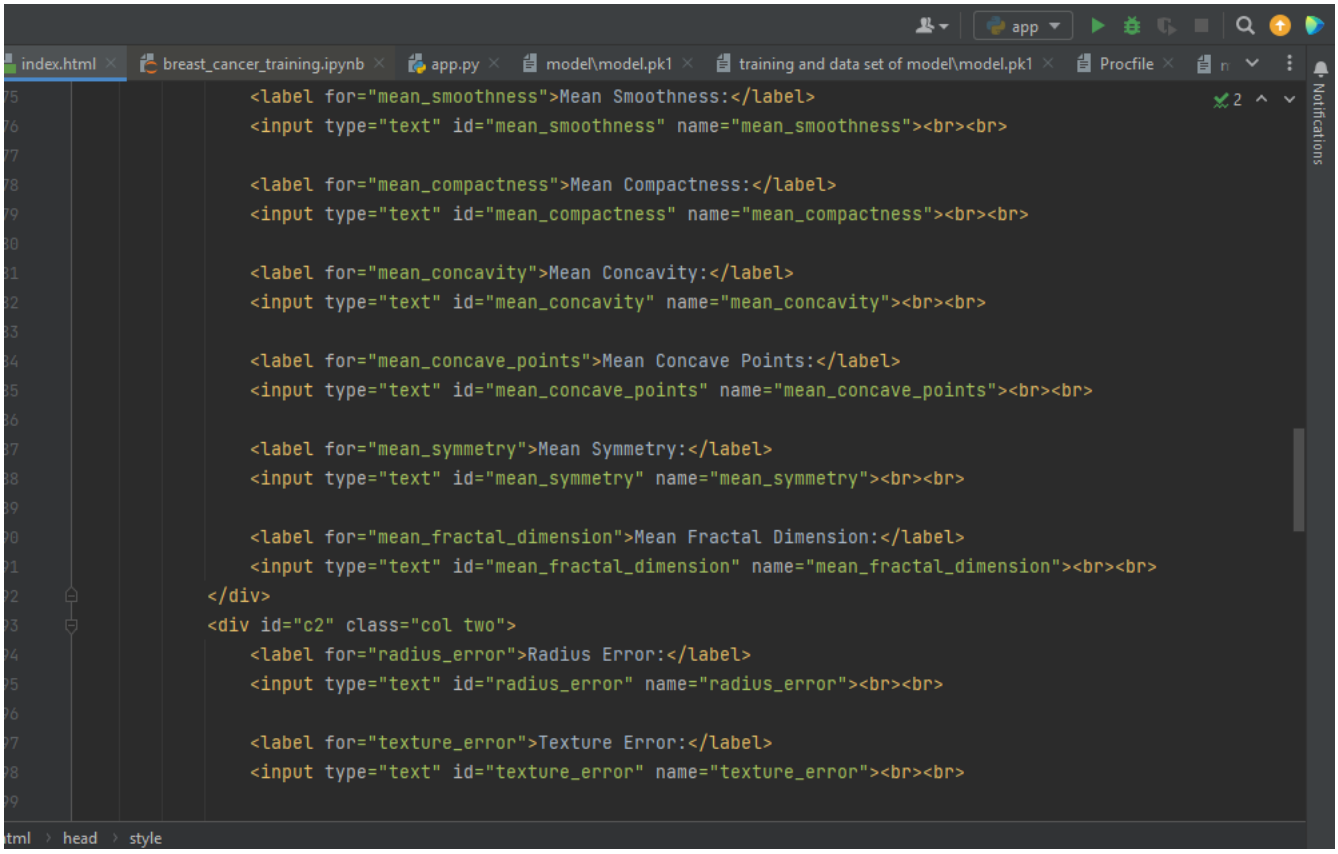
```python
pickle.dump(model,open("model.pk1",'wb'))
```

Python

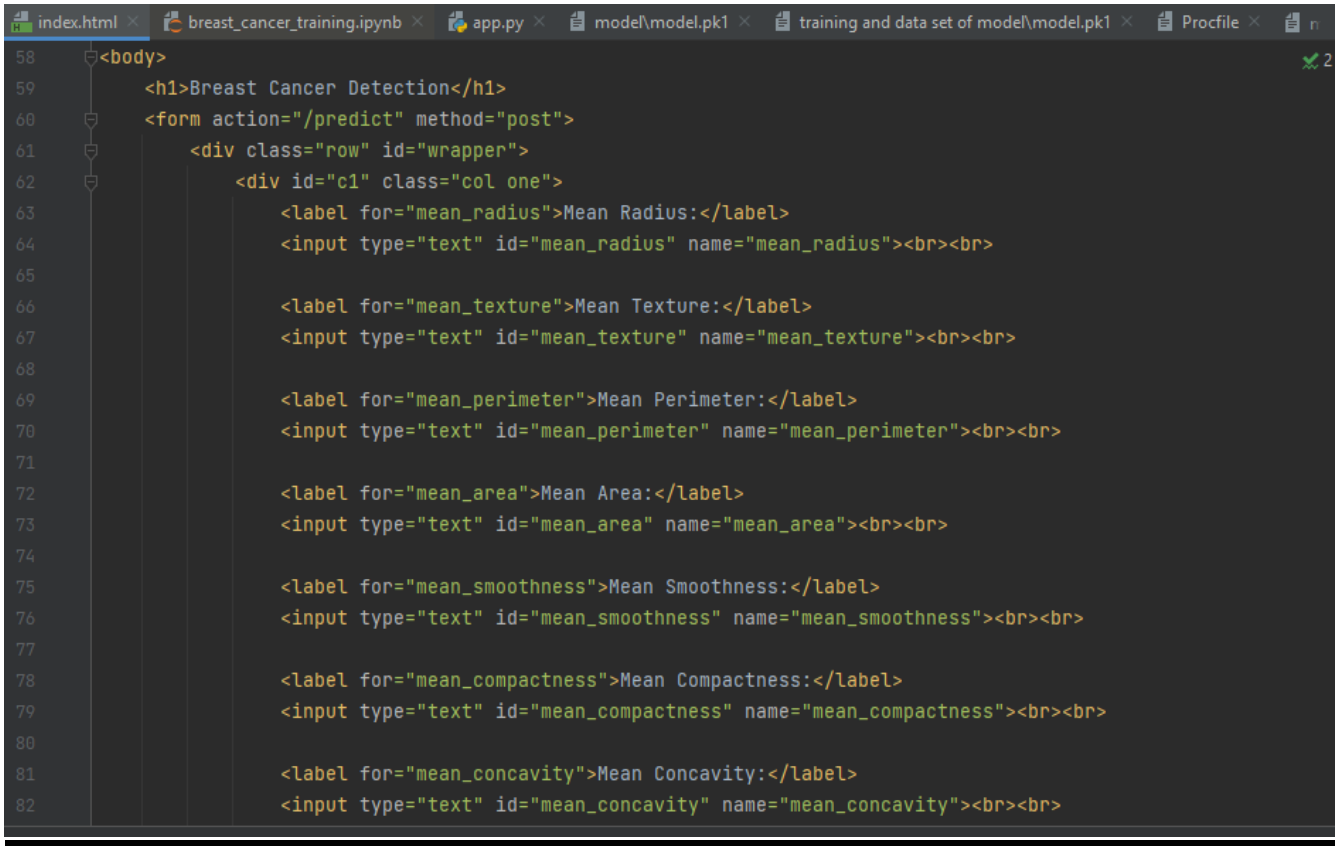
```python
data_frame.index()
```

5.1 RESULTS

Input: Now we will manually test whether the predictions are accurate or not by giving various criteria that are used to identify breast cancer as mentioned in the dataset as inputs.



```
            <label for="mean_smoothness">Mean Smoothness:</label>
            <input type="text" id="mean_smoothness" name="mean_smoothness"><br><br>

            <label for="mean_compactness">Mean Compactness:</label>
            <input type="text" id="mean_compactness" name="mean_compactness"><br><br>

            <label for="mean_concavity">Mean Concavity:</label>
            <input type="text" id="mean_concavity" name="mean_concavity"><br><br>

            <label for="mean_concave_points">Mean Concave Points:</label>
            <input type="text" id="mean_concave_points" name="mean_concave_points"><br><br>

            <label for="mean_symmetry">Mean Symmetry:</label>
            <input type="text" id="mean_symmetry" name="mean_symmetry"><br><br>

            <label for="mean_fractal_dimension">Mean Fractal Dimension:</label>
            <input type="text" id="mean_fractal_dimension" name="mean_fractal_dimension"><br><br>
        </div>
        <div id="c2" class="col two">
            <label for="radius_error">Radius Error:</label>
            <input type="text" id="radius_error" name="radius_error"><br><br>

            <label for="texture_error">Texture Error:</label>
            <input type="text" id="texture_error" name="texture_error"><br><br>
```



```
<body>
    <h1>Breast Cancer Detection</h1>
    <form action="/predict" method="post">
        <div class="row" id="wrapper">
            <div id="c1" class="col one">
                <label for="mean_radius">Mean Radius:</label>
                <input type="text" id="mean_radius" name="mean_radius"><br><br>

                <label for="mean_texture">Mean Texture:</label>
                <input type="text" id="mean_texture" name="mean_texture"><br><br>

                <label for="mean_perimeter">Mean Perimeter:</label>
                <input type="text" id="mean_perimeter" name="mean_perimeter"><br><br>

                <label for="mean_area">Mean Area:</label>
                <input type="text" id="mean_area" name="mean_area"><br><br>

                <label for="mean_smoothness">Mean Smoothness:</label>
                <input type="text" id="mean_smoothness" name="mean_smoothness"><br><br>

                <label for="mean_compactness">Mean Compactness:</label>
                <input type="text" id="mean_compactness" name="mean_compactness"><br><br>

                <label for="mean_concavity">Mean Concavity:</label>
                <input type="text" id="mean_concavity" name="mean_concavity"><br><br>
```

CHAPTER 6

TESTING

Various types of testing techniques can be employed to evaluate different aspects of the system. By employing different types of testing, a comprehensive evaluation of the breast cancer detection project can be conducted. This helps identify and address potential issues, improve the system's performance, and build confidence in its reliability and accuracy.

1. **Unit Testing:** Unit testing focuses on testing individual components or functions of the system. In the context of a breast cancer detection project, unit testing may involve testing specific functions or modules responsible for data preprocessing, feature extraction, or model training.
2. **Performance Testing:** Performance testing involves measuring the inference time of the model, analyzing memory consumption, and assessing the system's ability to handle larger datasets.
3. **Accuracy Testing:** Accuracy testing evaluates the accuracy of the breast cancer detection model in correctly classifying benign and malignant cases. It involves using an independent test dataset with known ground truth labels and comparing the model's predictions against the actual labels.
4. **Sensitivity Testing:** Sensitivity and specificity testing evaluates the model's ability to correctly identify true positive (sensitivity) and true negative (specificity) cases. It helps assess the model's performance in detecting malignant cases accurately while minimizing false positives.
5. **Robustness Testing:** Robustness testing evaluates the model's performance under different conditions, including variations in input data, noise, or perturbations.
6. **User Acceptance Testing:** It involves gathering feedback from medical professionals or end-users to assess the system's usability, user interface, and overall satisfaction.

CHAPTER 7

CONCLUSION AND FUTURE SCOPE

7.1 CONCLUSION

The breast cancer detection project using machine learning has demonstrated promising results in accurately identifying and classifying benign and malignant cases. By leveraging machine learning algorithms and techniques, such as feature extraction, model training, and prediction, the project has shown the potential to assist in early detection and diagnosis of breast cancer. The project's success in achieving high accuracy and performance highlights the effectiveness of machine learning in healthcare applications. In conclusion, the breast cancer detection project using machine learning has shown great potential for improving early detection and diagnosis of breast cancer. Continued research and development, along with collaboration between machine learning experts and healthcare professionals, can further enhance the project's impact and pave the way for more accurate, efficient, and accessible breast cancer detection systems.

7.2 FUTURE SCOPE

The breast cancer detection project using machine learning opens up several avenues for future exploration and enhancement.

- **Improving Model Performance:** Continuously refining and optimizing the machine learning model can help enhance its accuracy and sensitivity. Techniques such as hyperparameter tuning, feature selection, and ensembling can be explored to achieve even better results.
- **Incorporating Advanced Machine Learning Techniques:** Exploring advanced machine learning techniques, such as deep learning and convolutional neural networks (CNNs), can potentially improve the detection and classification accuracy. These techniques can effectively

handle complex image-based data, providing more comprehensive insights.

- **Integration with Imaging Technologies:** Integrating the breast cancer detection model with advanced imaging technologies, such as mammography, ultrasound, or MRI scans, can create a more comprehensive and accurate diagnostic system. Combining machine learning with imaging technologies can improve the precision and reliability of breast cancer detection.
- **Handling Imbalanced Data:** Addressing the challenge of imbalanced data distribution, where benign cases may significantly outnumber malignant cases, can be a focus for future research. Techniques such as oversampling, undersampling, or generating synthetic samples can help balance the dataset and improve model performance.
- **Real-Time Diagnosis:** Extending the project to real-time diagnosis and prediction can have significant clinical implications. Developing a system that can provide immediate predictions based on real-time data from patients can assist healthcare professionals in making timely and accurate decisions.
- **Validation and Clinical Trials:** Conducting extensive validation studies and clinical trials involving a diverse set of patient populations and healthcare settings can further validate the effectiveness and reliability of the breast cancer detection system. Collaboration with medical professionals and experts can help refine the system and assess its real-worldimpact.

APPENDIX

```
import numpy as np
import pickle
from flask import Flask, render_template, request

app = Flask(__name__)
model_filename='model.pkl'
with open(model_filename, 'rb') as f:
    model = pickle.load(f)

@app.route('/')
def home():
    return render_template('index.html')

@app.route('/predict', methods=['POST'])
def predict():
    # Read the values from the form as floats
    mean_radius = float(request.form['mean_radius'])
    mean_texture = float(request.form['mean_texture'])
    mean_perimeter = float(request.form['mean_perimeter'])
    mean_area = float(request.form['mean_area'])
    mean_smoothness = float(request.form['mean_smoothness'])
    mean_compactness = float(request.form['mean_compactness'])
    mean_concavity = float(request.form['mean_concavity'])
    mean_concave_points = float(request.form['mean_concave_points'])
    mean_symmetry = float(request.form['mean_symmetry'])
    mean_fractal_dimension = float(request.form['mean_fractal_dimension'])
    radius_error = float(request.form['radius_error'])
    texture_error = float(request.form['texture_error'])
    perimeter_error = float(request.form['perimeter_error'])
    area_error = float(request.form['area_error'])
    smoothness_error = float(request.form['smoothness_error'])
    compactness_error = float(request.form['compactness_error'])
    concavity_error = float(request.form['concavity_error'])
    concave_points_error = float(request.form['concave_points_error'])
    symmetry_error = float(request.form['symmetry_error'])
    fractal_dimension_error = float(request.form['fractal_dimension_error'])
    worst_radius = float(request.form['worst_radius'])
    worst_texture = float(request.form['worst_texture'])
    worst_perimeter = float(request.form['worst_perimeter'])
    worst_area = float(request.form['worst_area'])
    worst_smoothness = float(request.form['worst_smoothness'])
    worst_compactness = float(request.form['worst_compactness'])
    worst_concavity = float(request.form['worst_concavity'])
    worst_concave_points = float(request.form['worst_concave_points'])
    worst_symmetry = float(request.form['worst_symmetry'])
    worst_fractal_dimension = float(request.form['worst_fractal_dimension'])

    # Create a numpy array from the input values
    features = np.array([[mean_radius, mean_texture, mean_perimeter, mean_area,
mean_smoothness, mean_compactness,
                          mean_concavity, mean_concave_points, mean_symmetry,
mean_fractal_dimension, radius_error,
                          texture_error, perimeter_error, area_error,
smoothness_error, compactness_error,
                          concavity_error, concave_points_error, symmetry_error,
fractal_dimension_error,
                          worst_radius, worst_texture, worst_perimeter, worst_area,
worst_smoothness,
                          worst_compactness, worst_concavity, worst_concave_points,
worst_symmetry,
                          worst_fractal_dimension]])
```

```
    # Perform prediction using the loaded model
    prediction = model.predict(features)


    # Convert the prediction (0 or 1) to a meaningful result
    if prediction[0] == 0:
        result = 'Malignant'
    else:
        result = 'Benign'

    return render_template('result.html', prediction=result)
if __name__ == '__main__':
    app.run(debug=True)
```

REFERENCES

- https://www.kaggle.com/code/vikasukani/breast-cancer-prediction-using-machine-learning
- https://www.ibm.com/topics/exploratory-data-analysis
- https://en.wikipedia.org/wiki/Bivariate_analysis#:~:text=Bivariate%20analysis%20is%20one%20of,the%20empirical%20relationship%20between%20them.
- https://www.simplilearn.com/tutorials/machine-learning-tutorial/feature-selection-in-machine-learning#:~:text=EdgeJoin%20Cohort-,What%20is%20Feature%20Selection%3F,you%20are%20trying%20to%20solve.
- https://www.sciencedirect.com/topics/engineering/confusion-matrix#:~:text=A%20confusion%20matrix%20is%20a,performance%20of%20a%20classification%20algorithm.
- https://www.javatpoint.com/k-nearest-neighbor-algorithm-for-machine-learning
- https://www.javatpoint.com/machine-learning-support-vector-machine-algorithmhttps://www.tutorialspoint.com/sdlc/sdlc_waterfall_model.h