

**breast cancer detection**

**Using AdaBoost Classifier**



**Gayathri R**

**B Krishna Raja Sree**

**NAVEEN S**

**Shaik Ayesha Parveen**

**Joseph Boban**

SuBmitted by: team 1

-

**INDEX**

|  |
| --- |
| 1. Introduction to Breast Cancer and Medical Diagnosis  -3 |
| 2. Dataset Information -3 |
| 3. Data Cleaning and Preprocessing -4 |
| 4. Analysis by Plotting Graphs   * Bar Chart -7 * Pie Chart * Heat Map * Pair-Plot |
| 5. Exploratory Data Analysis (EDA)   * Outlier Detection - 10 * Detection Methods * Distribution of Features |
| 6. Data Processing Steps -18 |
| 7. Model Building and Evaluation -23 |
| 9. Hyperparameter Tuning  -33 |
| 10.Feature Selection for the model -35 |
| 11.The final Results -36 |
| 12. Stream lit Application - AI Breast Cancer Diagnostic Assistant -38 |

**List of Tables:**

1. Table of Dataset Characteristics
2. Summary Statistics for Numerical Columns
3. Model Performance Metrics

**List of Graphs:**

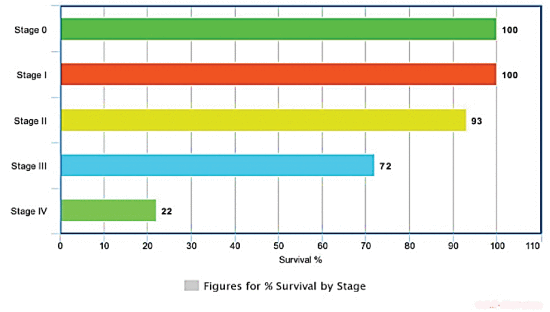
1. Bar Chart: Count of Benign and Malignant Cases
2. Pie Chart: Distribution of Diagnoses
3. Heat Map: Correlation Matrix
4. Pair-Plot: Feature Relationships
5. ROC Curve
6. Precision-Recall Curve
7. Confusion Matrix
8. Classification Report Visualization

**Introduction to Breast Cancer and Medical Diagnosis**

Breast cancer is one of the most common cancers affecting women worldwide, contributing significantly to cancer-related deaths. Early detection is crucial for improving survival rates, as timely treatment can lead to better outcomes.

Traditional breast cancer diagnosis relies on physical exams, imaging techniques like mammograms, and laboratory tests such as biopsies. While effective, these methods can be expensive, time-consuming, and occasionally prone to errors.

Machine learning offers a modern approach to assist in breast cancer detection by analyzing large medical datasets to identify patterns and make predictions. This project uses machine learning techniques to develop a reliable model for detecting breast cancer early, helping improve accuracy and supporting healthcare systems in providing better patient care.



**Dataset Information**

The UCI Breast Cancer Wisconsin Diagnostic Dataset is a widely used open-source dataset for breast cancer research and diagnostics. It contains detailed measurements of cell features, including size, shape, texture, and more, extracted from digitized images of fine needle aspirates of breast masses.

The experimental results have been demonstrated with the help of the dataset  
Link: <https://archive.ics.uci.edu/dataset/17/breast+cancer+wisconsin+diagnostic>

**Introduction**

The **Diagnostic Wisconsin Breast Cancer Database** is a multivariate dataset widely used in the health and medicine domain for research and diagnostic purposes. It is designed for **classification tasks**, helping differentiate between benign and malignant tumors. The dataset consists of **569 instances** and **30 real-valued features** that describe various physical characteristics of cell nuclei, such as radius, texture, perimeter, area, and smoothness. Derived from fine needle aspirate images of breast tissue, these features provide valuable insights into tumor properties. Its structured nature and relevance to medical diagnostics make it a benchmark for developing machine learning models in breast cancer detection.

**1. Data Cleaning and Preprocessing**

* **shape** - dimensions of the dataset, providing the number of rows and columns as a tuple.

The dataset consists of 569 rows and 32 columns.



* **Columns** and their names

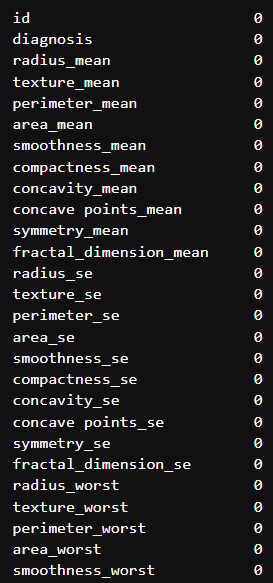
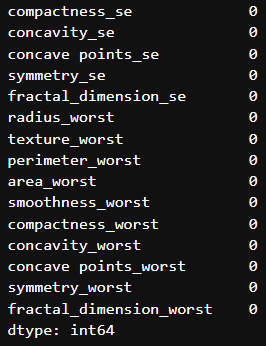
The dataset consists of 32 columns, out of which 31 are numeric (int64 and float64) and 1 is categorical (object) variables.

A screen shot of a computer code

Description automatically generated

* Null values represent **missing or incomplete data** in the dataset, which can affect the accuracy and performance of the AI model.

There are **no missing values** in the Dataset.

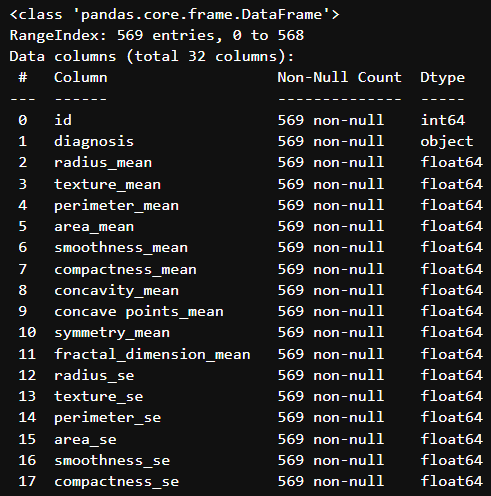
 

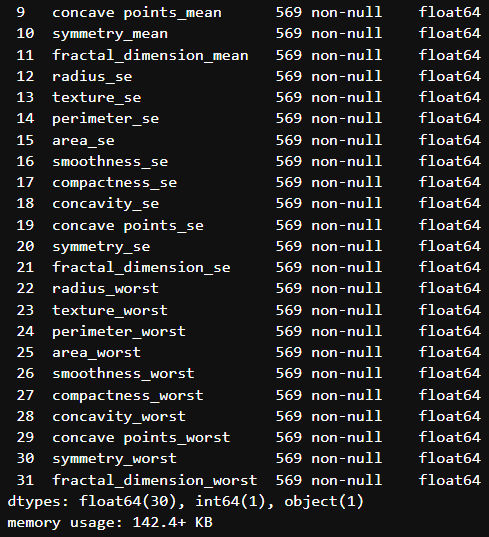
* The dataset contains **no duplicate rows**, ensuring 100% unique data for accurate analysis.



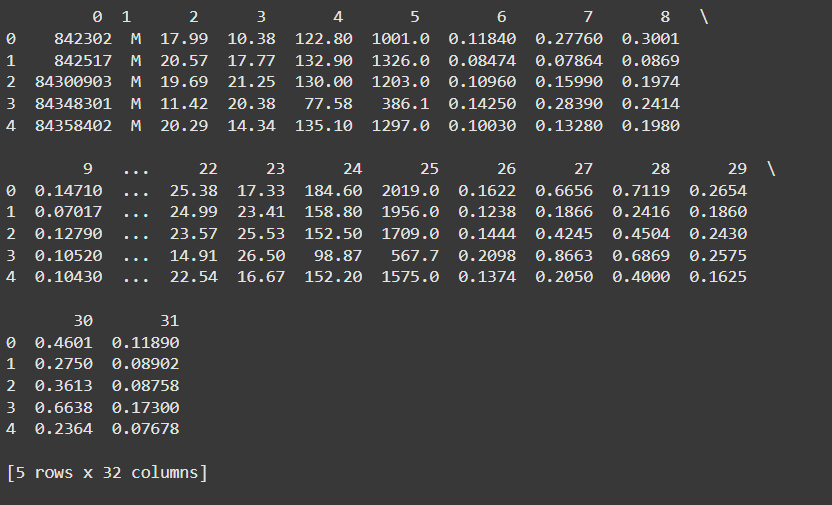
* **info()** - provides a quick summary of a DataFrame. It is a useful tool for understanding the structure and quality of the dataset at a glance.

The number of rows and columns, column names, their data types, the count of non-null values in each column, and the memory usage is displayed.

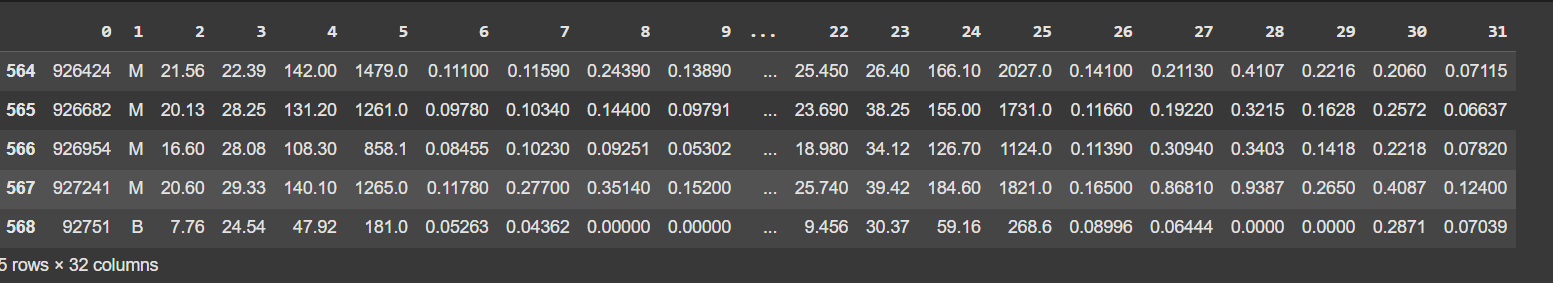




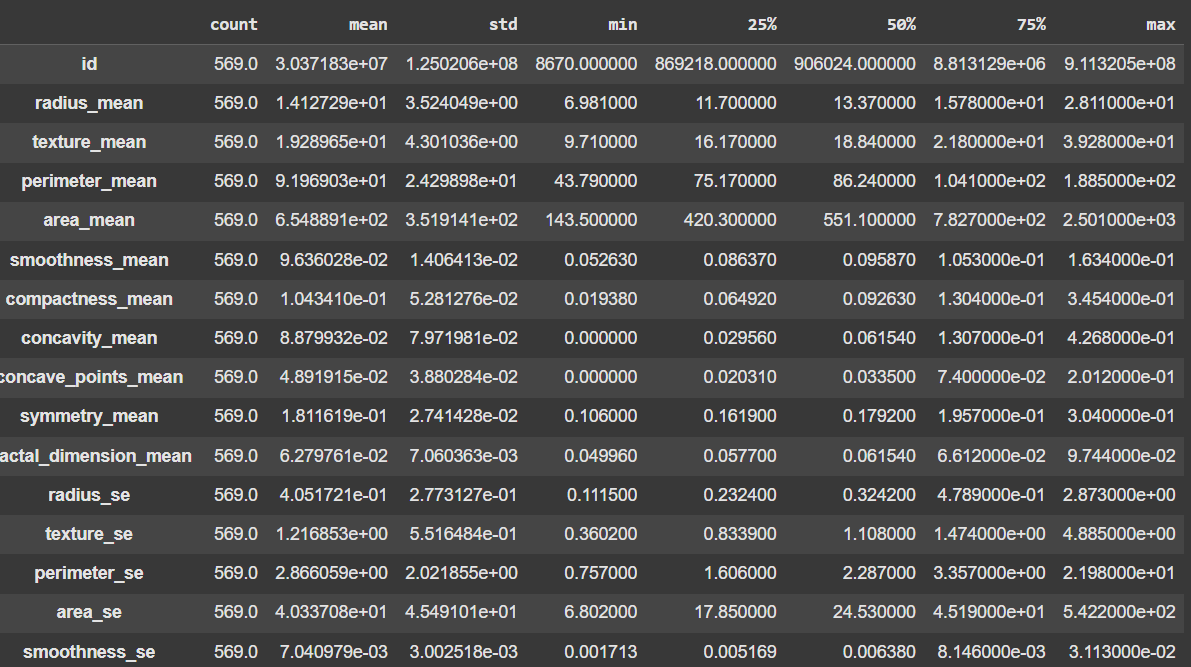
* **head -** Displays the **first five rows** of the dataset, giving a quick preview of the initial records and structure of the data.

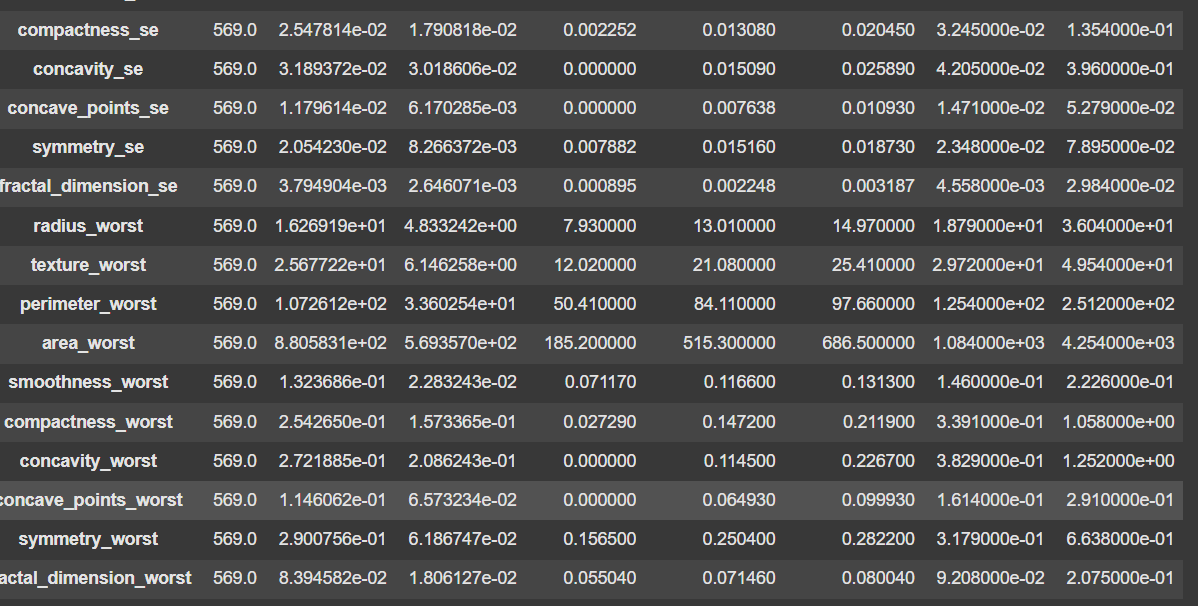


* **tail** - Shows the **last five rows** of the dataset, providing a glimpse of the final records and structure of the data.



* **describe().T** - generates summary statistics for numerical columns in the dataset and transposes the result making it easier to interpret the data.



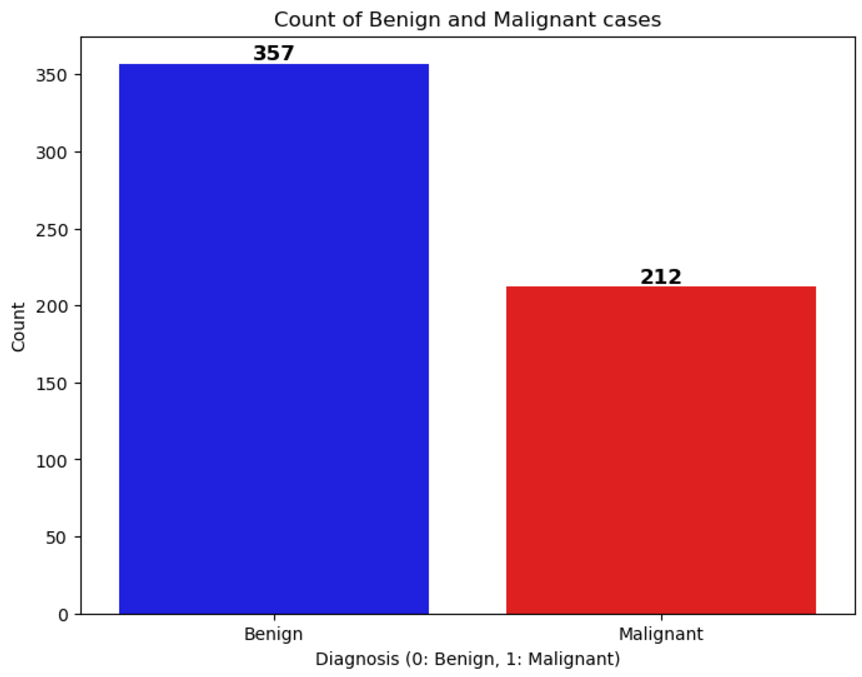


**2. Analysis by Plotting Graphs**

**Bar Chart**

The ‘Count of Benign and Malignant cases’ represents the distribution of breast cancer diagnoses in the dataset. Here, M (Malignant) is encoded as 1, and B (Benign) as 0. This plot helps visualize the proportion of malignant and benign cases, providing insights into the class balance, which is crucial for model performance evaluation.

The counts of benign and malignant cases are 357 and 212 respectively, and in terms of percentage, they are 62.7% and 37.3%.



**Pie Chart**

A pie chart is a circular graph that shows how different categories contribute to a whole. Each slice represents a category and its size shows its proportion to the total. Pie charts are useful for displaying simple, proportional data.

Pie charts are appropriate for showing the relationship between parts and a whole for categorical data. They can be used to visualize how each subset contributes to the entire dataset

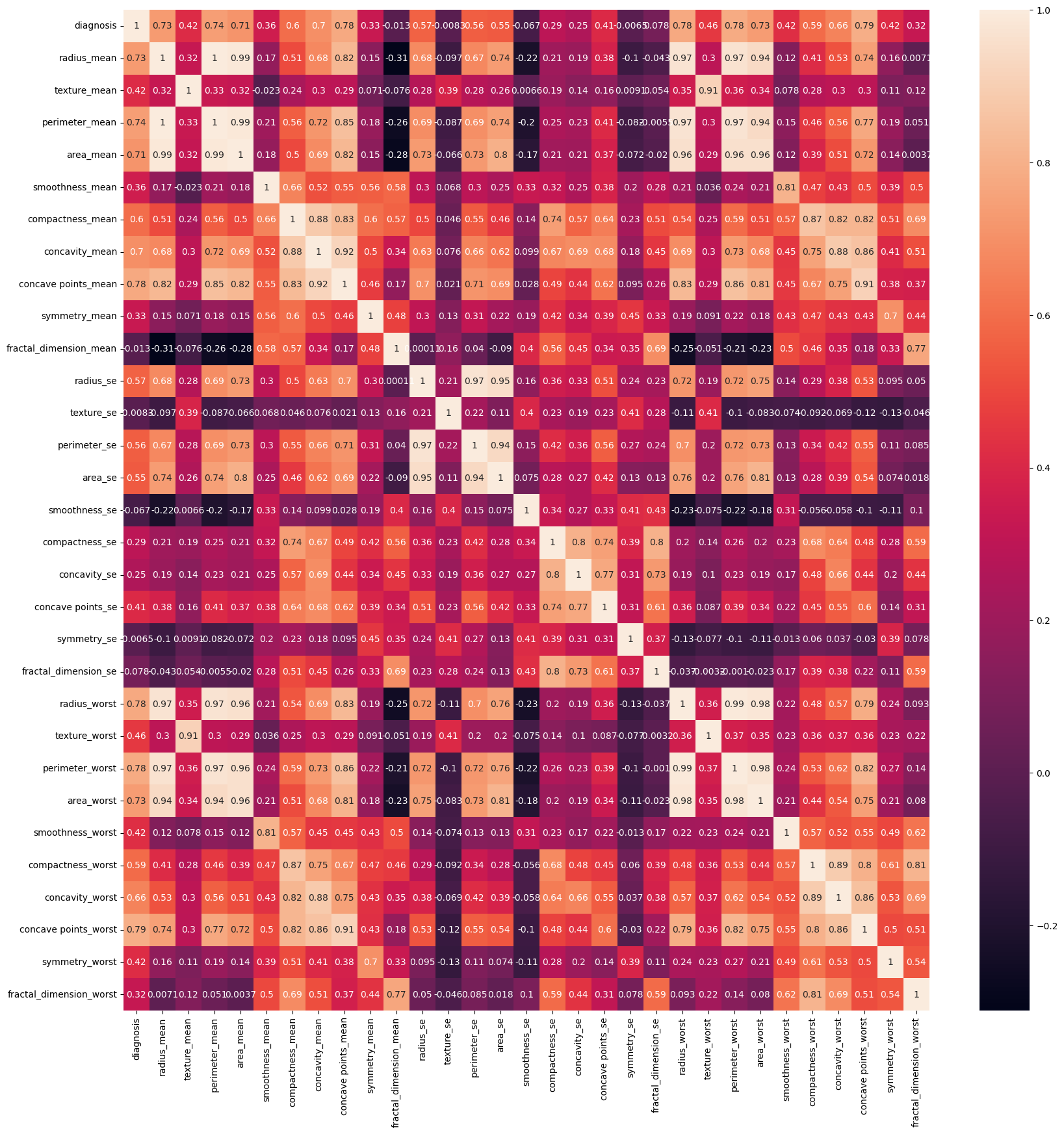
A blue and red circle with text

Description automatically generated

**Heat Map**

A heatmap is a graphical representation of data where values are depicted using varying colors. In data analysis, it is commonly used to **visualize correlation matrices**, showing the relationships between variables.

The intensity of the colour indicates the strength of the correlation, with warmer colors (e.g., red) typically representing higher positive correlations and cooler colors (e.g., blue) representing negative correlations. Heatmaps are valuable for identifying patterns, trends, and multicollinearity in datasets.

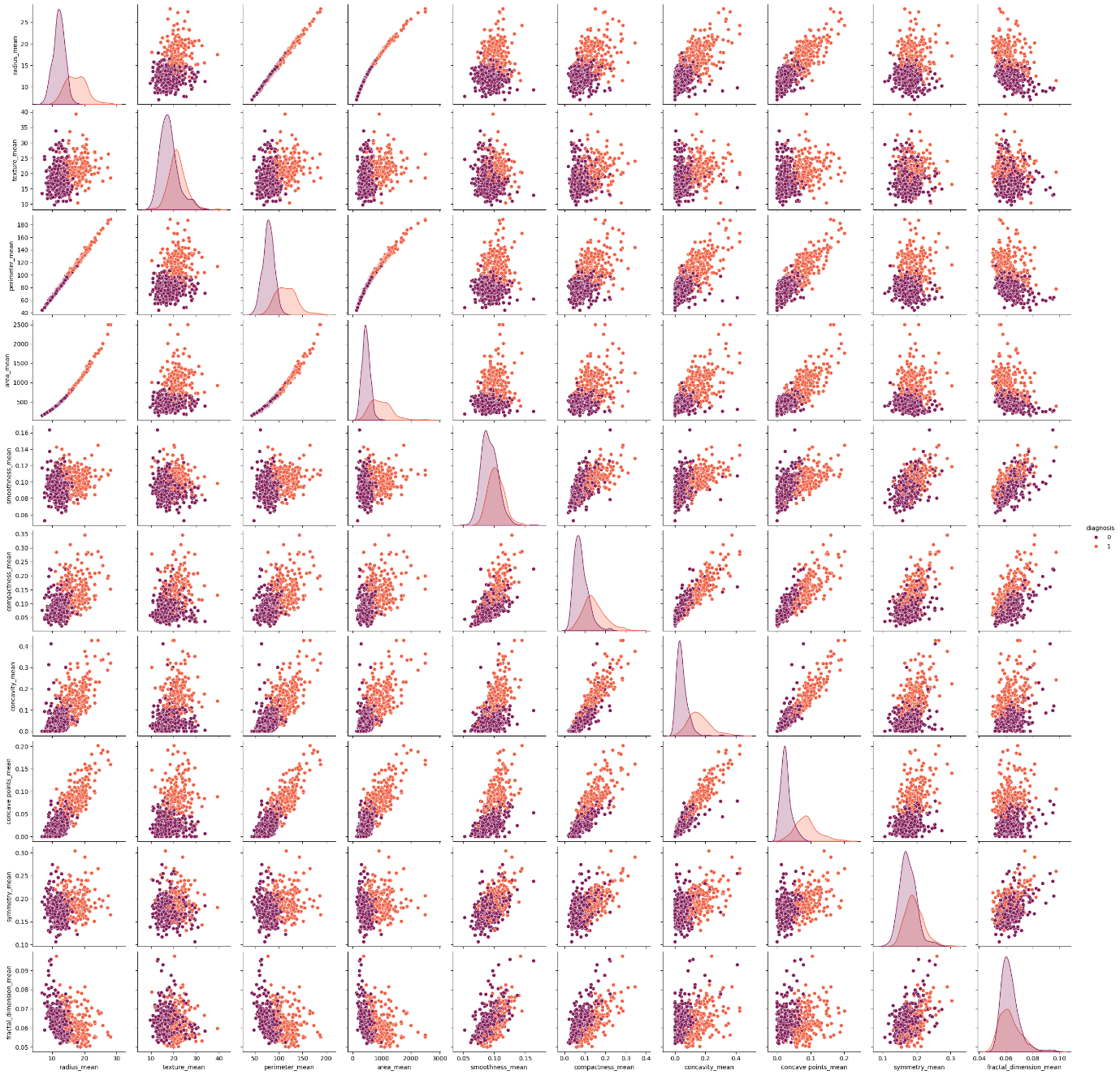


**Fig:** Heat Map

**Pair-Plot**

A pairplot is a **matrix of scatter plots** that visualizes relationships between multiple numerical features in a dataset. Each plot in the pairplot shows the **relationship between two variables**, while the diagonal often displays the distribution of individual features (e.g., histograms).

This helps identify patterns, correlations, and potential outliers between the features. Pairplots are particularly useful for detecting linear or non-linear relationships, understanding feature interactions, and assessing data quality across different pairs of variables.

****

**Fig:** Pair-Plot

**3. Exploratory Data Analysis (EDA)**

Exploratory Data Analysis (EDA) is the process of analyzing and visualizing a dataset to summarize its main characteristics, often using statistical and graphical techniques. The goal of EDA is to uncover patterns, identify anomalies, test hypotheses, and check assumptions before applying machine learning models. It helps to better understand the data, clean it, and prepare it for further analysis or modeling. Common EDA tasks include examining distributions, relationships between variables, missing values, and outliers using tools like histograms, scatter plots, box plots, and correlation matrices**.**

**Outlier Detection**

**Outlier** - are data points that significantly differ from other observations in a dataset. They can be much higher or lower than most of the other values, and they may indicate variability in the data, i.e., errors.

**Detection Methods:**

* **Visual methods**: Boxplots, histograms, and scatter plots can be used to visually spot outliers.
* **Statistical methods**:
  + **Z-score**: Identifies how far a data point is from the mean, measured in standard deviations.
  + **IQR (Interquartile Range)**: Identifies outliers by looking at the spread of the middle 50% of the data and finding values beyond a specific range.

**IQR Method:**

**Steps to Calculate the IQR:**

1. **Sort the data**: Arrange the data in ascending order.
2. **Find the Quartiles**:
   * **Q1 (First Quartile)**: The 25th percentile of the data, which is the value below which 25% of the data points fall.
   * **Q3 (Third Quartile)**: The 75th percentile of the data, which is the value below which 75% of the data points fall.

The IQR is calculated as



1. **Determine the Outlier Boundaries**:

* The **Lower Bound** is calculated as



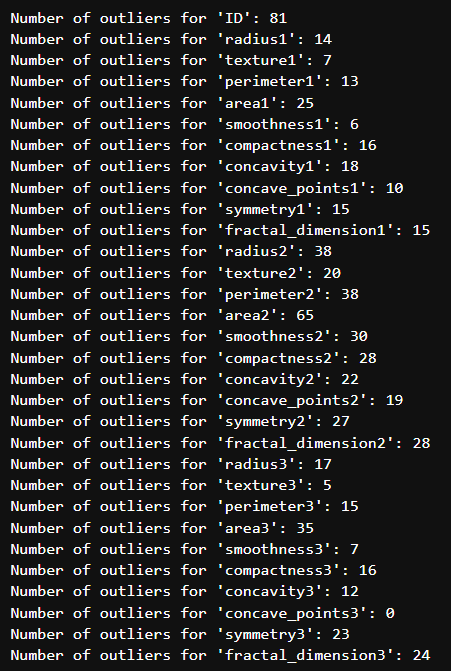
* The **Upper Bound** is calculated as



1. **Identify Outliers**:

* Any data points below the **Lower Bound** or above the **Upper Bound** are considered outliers.

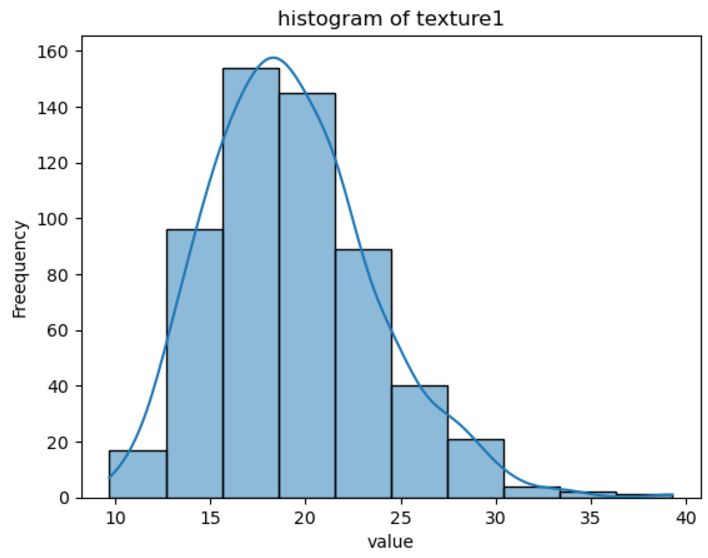
Number of Outliers for each features



A graph of a graph

Description automatically generated

**Distribution of Different Features**

 A graph of a normal distribution

Description automatically generated

A graph of a graph of value

Description automatically generated A graph of a normal distribution

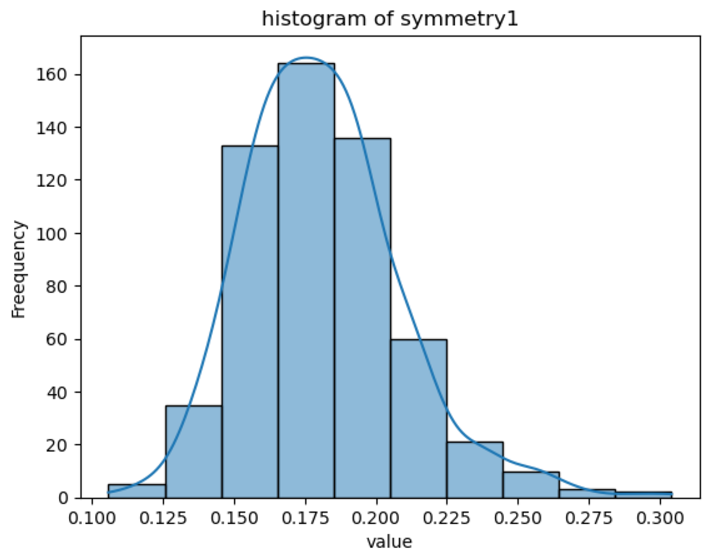
Description automatically generated

A graph of a graph

Description automatically generated A graph of a graph

Description automatically generated

A graph of a graph

Description automatically generated 

A graph of a graph with a line

Description automatically generatedA graph of value and value

Description automatically generated

A graph of value and value

Description automatically generated A graph of value and value

Description automatically generated

A graph of a graph

Description automatically generated A graph of a graph

Description automatically generated

A graph of a graph

Description automatically generated A graph with a line graph

Description automatically generated

A graph of a graph

Description automatically generated A graph of a line graph

Description automatically generated

A graph of a graph

Description automatically generated A graph of a graph

Description automatically generated

A graph of value and value

Description automatically generated A graph of value and value

Description automatically generated

A graph of value and value

Description automatically generated A graph of a normal distribution

Description automatically generated

A graph of a normal distribution

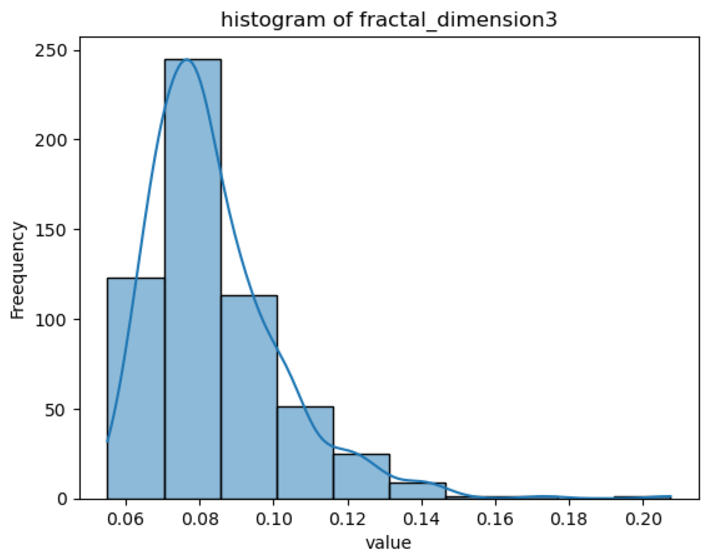
Description automatically generated A graph of a graph

Description automatically generated

A graph of a graph of value

Description automatically generated with medium confidence A graph of a line graph

Description automatically generated



The Distributions are either near normal or right skewed, therefore we will be using z scaling instead of log transformation and min-max scaling

**6.Data Processing Steps**

* Dropping ID column as it doesn't carry any meaningful information.



Checking whether ID column is dropped and displaying the count of the columns.

A screen shot of a computer screen

Description automatically generated

Number of columns has reduced to 31.



* Encoding values B and M

Machine learning models typically work with numerical data, not categorical or string-based values. Encoding **B** and **M** as 0 and 1, converts these categorical labels into a numerical format that algorithms can process effectively.

A computer screen shot of a black screen

Description automatically generated



The values of diagnosis – B and M are encoded as 0 and 1.

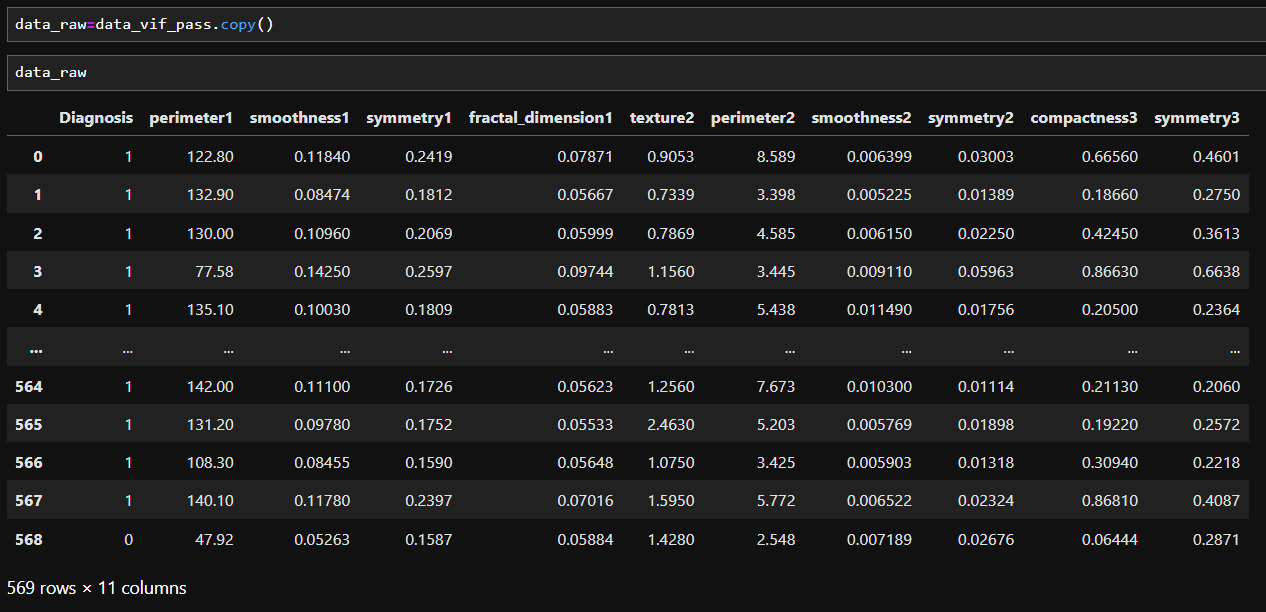
**A screenshot of a computer

Description automatically generated**

A screenshot of a computer

Description automatically generated

* We have data imbalance, therefore we will be using smote, but to compare we need the initial version, therefore we will create two copies, one for smote and one for the original data



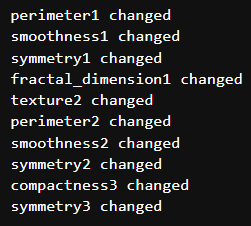
**Creating four versions of the code so that we could compare the 4 versions**

(outlier smoothed and raw) x (with or without smote)

The smote version here is a data copy for the smote activity, it won’t be directly used, a new variable will be defined after applying smote and that dataframe will be passed to build the model.

Similarly for the smoothed data.

**Smoothing the raw data**



**Checking the number of outliers of the smoothed data**

A screenshot of a computer screen

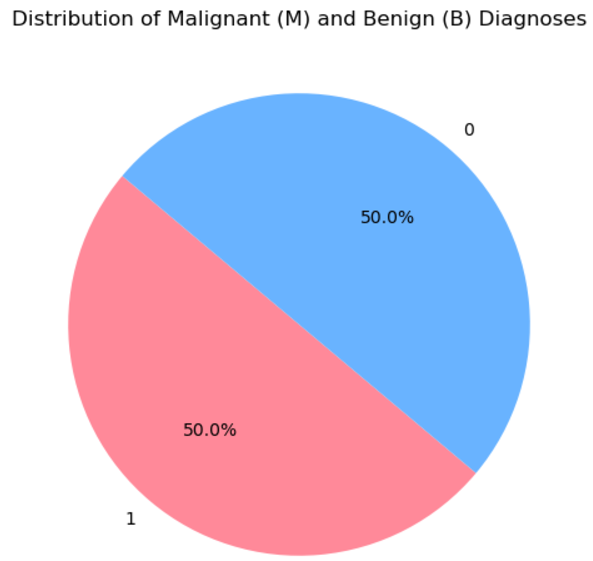
Description automatically generated

A screen shot of a computer screen

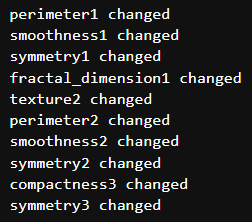
Description automatically generated

**Using Smote for unbalanced data**

For smote to be implemented, it's better not to go to the function we will be defining later for handling smote data as the data splitting is happening inside the function, so implementing it here, so we are doing the smote first and then going to the function.



Creating **two versions** after smote so that we can smooth one version of it and keep the other in raw smote form



Creating a function to get the analysis, scaling and the other process in a single go, it will help us to compare multiple versions of the data with multiplier classifier and will help to choose the right one by reducing manual iterations of code.

inside the function, data splitting, data scaling, model fitting, and evaluation will be done, and the evaluation report will be done.

**7.Model Building and Evaluation**

**Part 1: Without SMOTE**

A screenshot of a computer

Description automatically generated

A graph of a curve

Description automatically generated

A screen shot of a black screen

Description automatically generated

A graph of a curve

Description automatically generated

A screenshot of a computer

Description automatically generated

A graph of a tree model

Description automatically generated

A screenshot of a computer

Description automatically generated

A graph with a line and a red line

Description automatically generated with medium confidence

A screen shot of a black screen

Description automatically generated

A graph of a curve

Description automatically generated

**Part 2:With Smote**

A screenshot of a computer screen

Description automatically generated

A graph of a curve

Description automatically generated

A screenshot of a computer

Description automatically generated

A graph with a line and a red line

Description automatically generated

A screenshot of a computer

Description automatically generated

A graph of a tree model

Description automatically generated

A screenshot of a computer

Description automatically generated

A graph with a line and a red line

Description automatically generated

A screenshot of a computer

Description automatically generated

A graph of a model

Description automatically generated with medium confidence

Analysing the two versions, we can clearly infer that the best result is getting from AdaBoost using smote data.

Now doing the hyperparameter tuning using GridSearchCV.

**8.Hyperparameter Tuning**

Using GridSearchCV we are trying to find the best hyperparameters so that we can fine tune our model further.

* Implementing GridSearchCV



* Initialize AdaBoostClassifier with the best hyperparameters

**Implementing GridSearchCV for data with SMOTE**

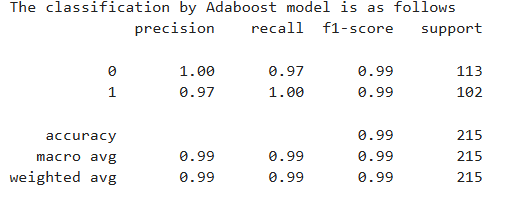
****

**Implementing GridSearchCV for data without SMOTE**

****

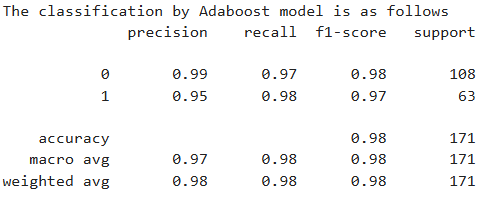
These are the hyperparameters we got for both versions of data, no trying to build the model using this

**Performance Analysis using AdaBoost Model after hyperparameter tuning with SMOTE**



Improved accuracy to 99%, with excellent precision and recall.

**Performance Analysis using AdaBoost Model after hyperparameter tuning without SMOTE**



Improved accuracy to 98%, with excellent precision and recall.

**Final Model Choice:**

**AdaBoost with SMOTE** is chosen as the final model due to its excellent performance on both balanced and imbalanced datasets, aided by hyperparameter tuning.

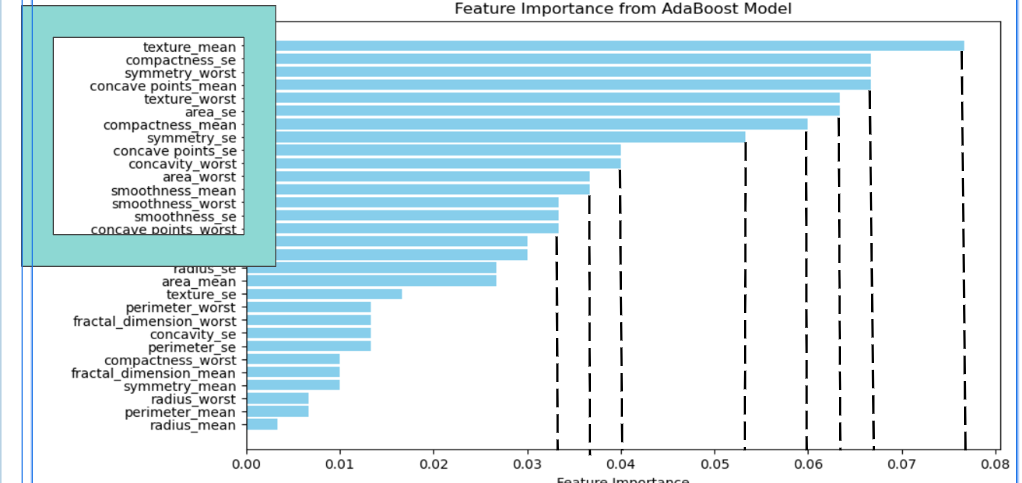
**9.Feature Selection for the model**

To create the model, using 32 input variable is almost impossible, so it’s better to select the most important features only, for that we are using the Feature Importance facility of AdaBoost.

We are applying to both the versions, smote and without smote though we are finalized which model to choose, for an additional confirmation.

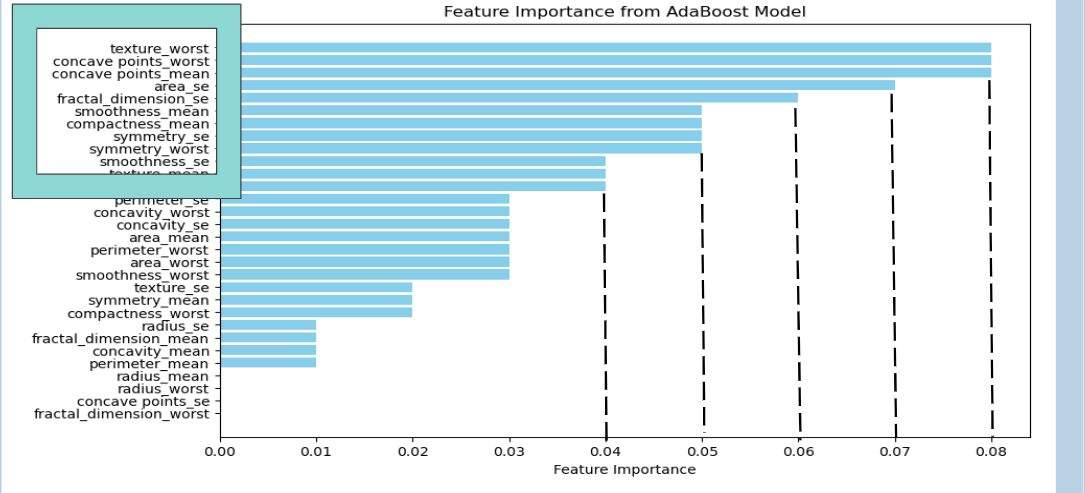
A Cut off of .03 is used for Variable selection

**Data without Smote**



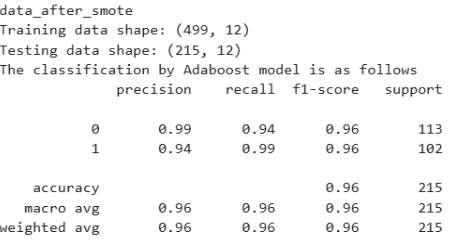
The features in the blue box has crossed the cutoff and are selected for the final model building test.

**Data with Smote**

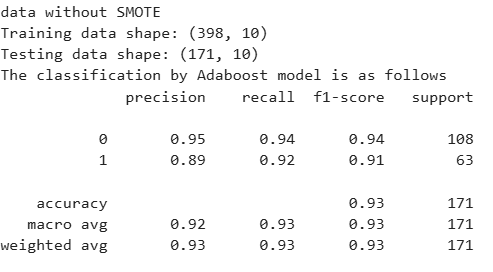


Now we are building the models, using the AdaBoost Algorithm and the hyperparameter tuning is also done simultaneously as we don’t have to do it in different steps.

**11.The final Results**



This is for SMOTE data, we can see the model performance is significantly good and it is having some very good recall for Malignant Cancer which is crucial in predictions using medical data.



For data without Smote the accuracy is just 92 and the recall is just 92, which is not acceptable in this case.

Considering both the cases we are going to build the Streamlit app using SMOTE version of the data.

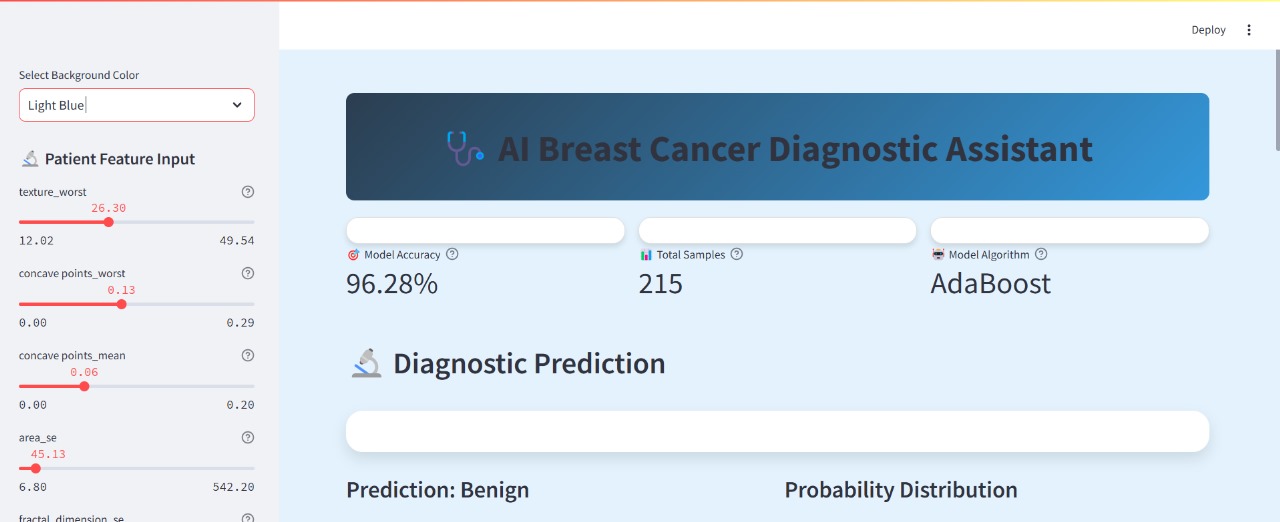
For the Streamlit app, the data needed Is the model, weights, features, Model Strength Matrices such as accuracy, Precision etc.

The data needed is saved as Pickle file(.pkl) and it will be accessed by the Streamlit app

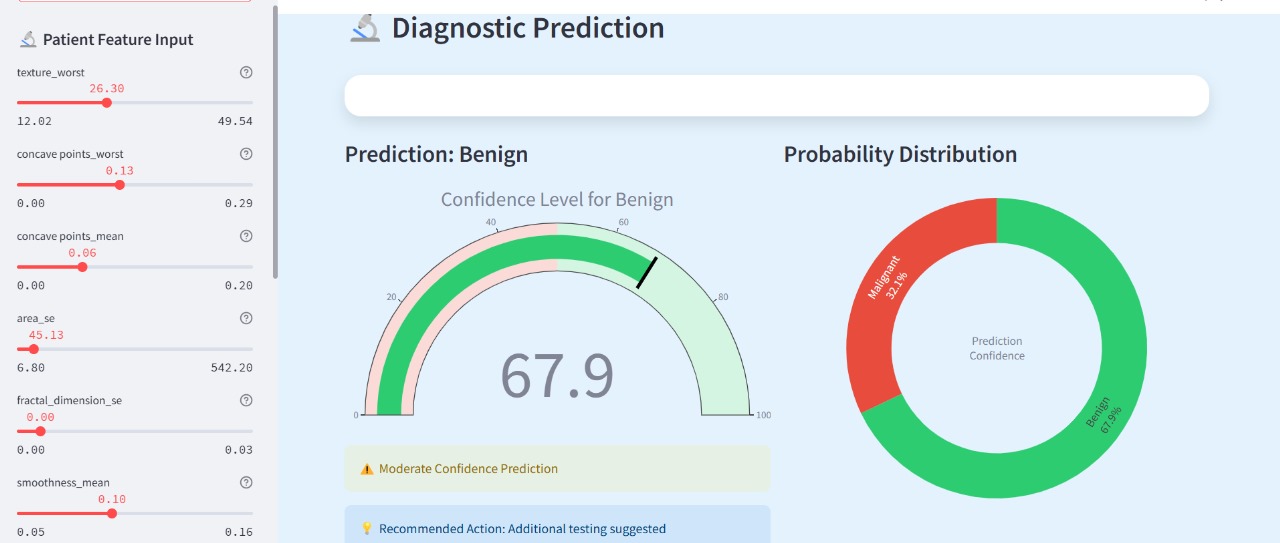
**12.Stream lit Application**

**-AI BREAST CANCER DIAGNOSTIC ASSISTANT**

The AI Breast Cancer Diagnostic Assistant is a powerful tool designed to aid in the diagnosis of breast cancer using patient input features. Utilizing the AdaBoost algorithm, the model achieves an impressive accuracy of 95.32% based on 171 samples. This tool enables healthcare professionals to make accurate and timely diagnostic decisions, significantly enhancing clinical efficiency and patient care.

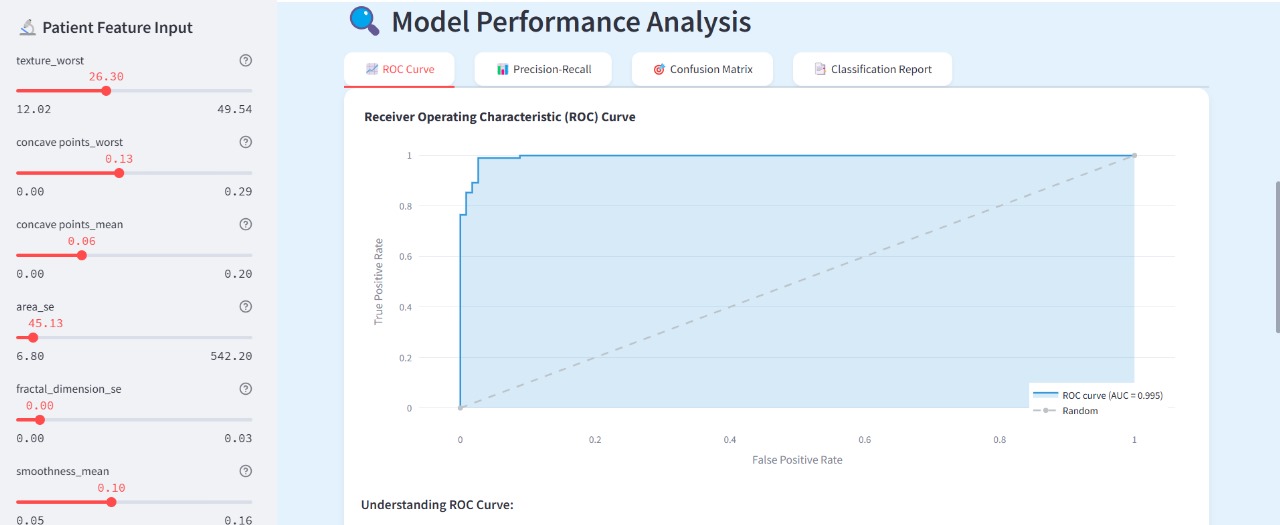


A diagnostic prediction tool for medical analysis, which predicts whether a condition is benign or malignant. The current prediction is "Benign" with a confidence level of 67.9%, marked as moderate confidence. The probability distribution pie chart indicates a 32.1% chance for malignancy. A recommended action suggests additional testing due to moderate confidence.

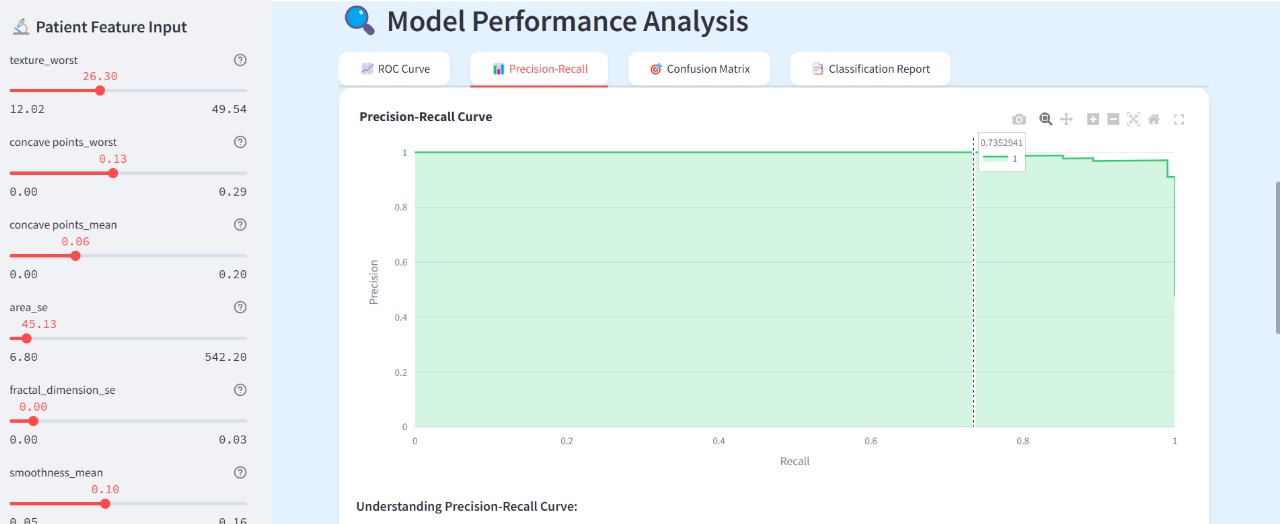


**MODEL PERFORMANCE ANALYSIS**

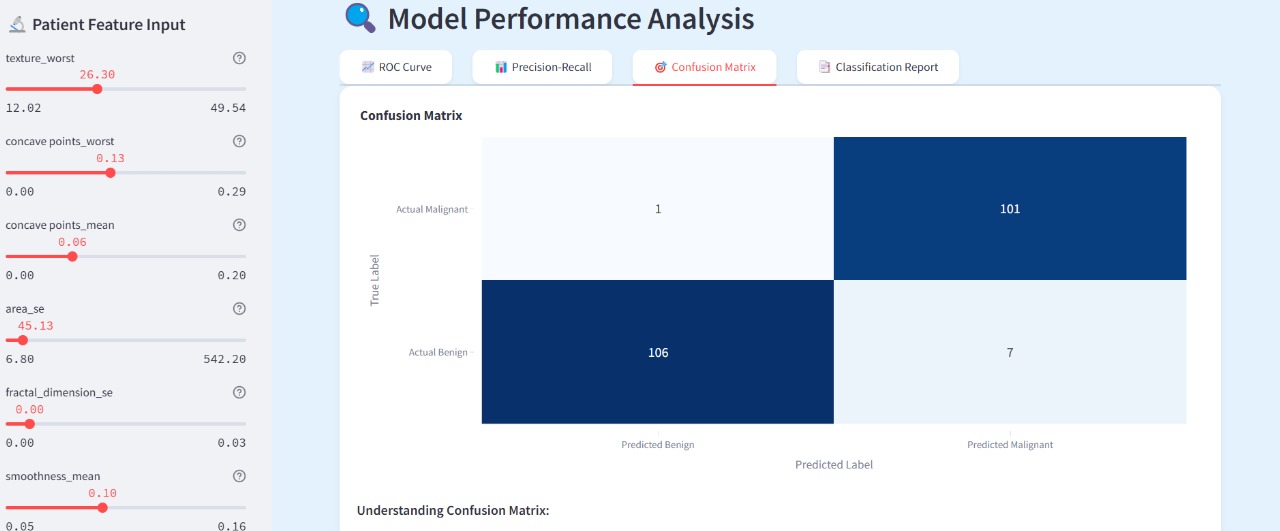
* 1. ROC CURVE



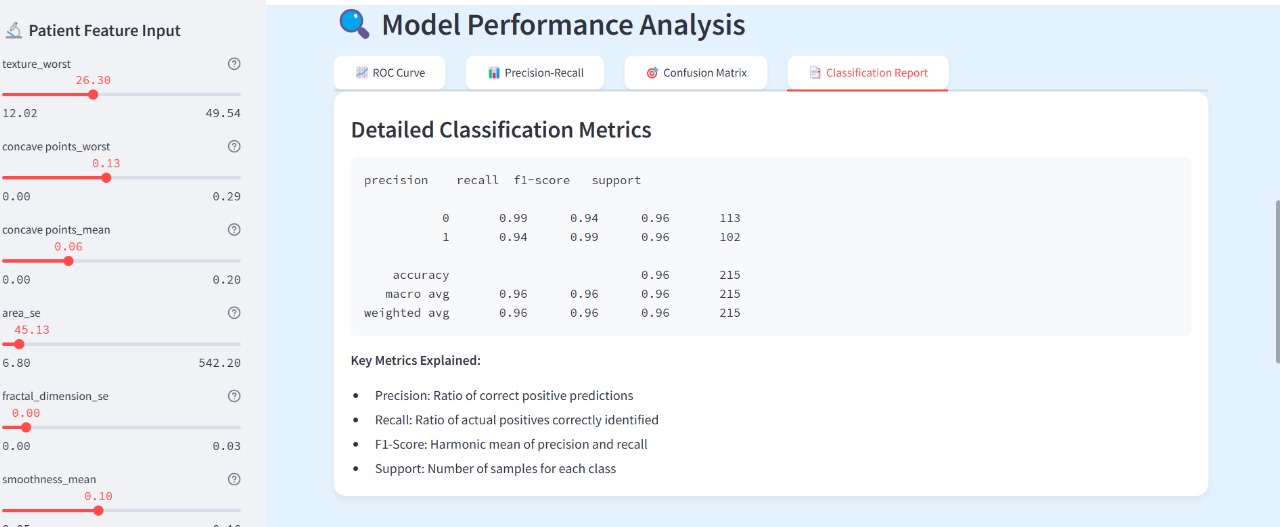
* 1. PRECISION-RECALL

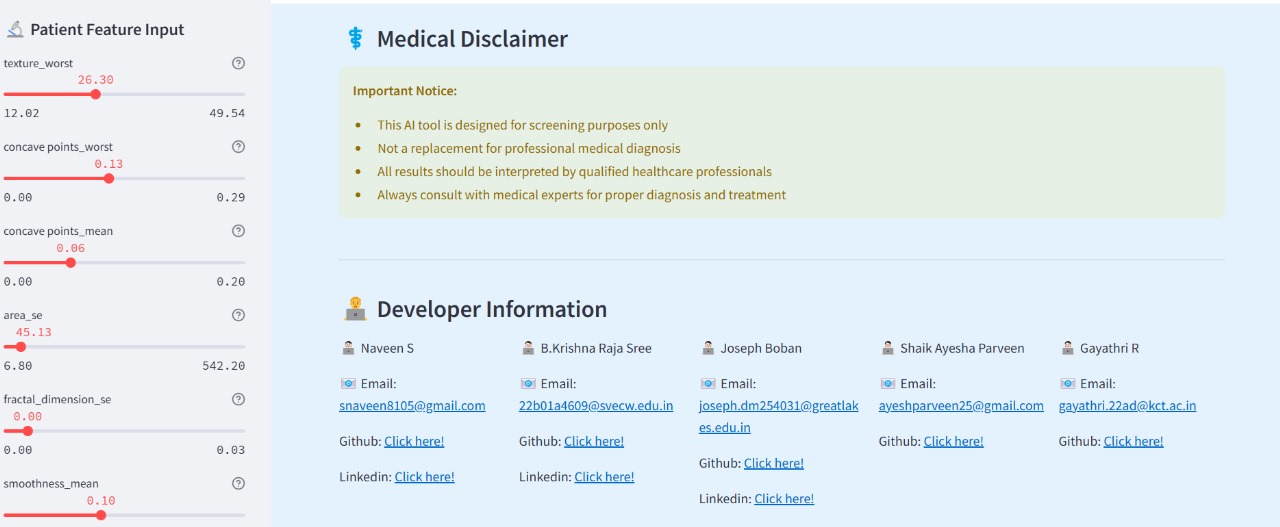


* 1. CONFUSION MATRIX



* 1. CLASSIFICATION REPORT





The app has a Graphical User interface, it has provided sliders for each feature, the model will run automatically without any run command. On that the confidence of prediction, probability distribution and Confidence is displayed and the current model used for the prediction is shown and its performance matrices also, if the model is updated, it is able to accept the new model and update the performance to the new model.