**NAMIBIA UNIVERSITY OF SCIENCE AND TECHNOLOGY**

Faculty of Computing and Informatics

School of Computing

Department of Software Engineering

DTA621 Data Analytics, SEMESTER 4, 2024

DTA621 Project: Breast Cancer Recurrence Prediction Report



Table of Contents

[Group Member List: 3](#_Toc179510367)

[Introduction 4](#_Toc179510368)

[Problem Statement: 4](#_Toc179510369)

[1. Column Understanding and Data Types: 4](#_Toc179510370)

[2. Data Cleaning: 6](#_Toc179510371)

[2.1. Handling Missing Values 6](#_Toc179510372)

[2.2. Verification of Missing Value Handling 7](#_Toc179510373)

[2.3. Final Data Inspection 7](#_Toc179510374)

[3. Exploratory Data: 7](#_Toc179510375)

[3.1. Distribution of Breast Cancer Class 7](#_Toc179510376)

[3.2. Tumor Size vs. Degree of Malignancy 8](#_Toc179510377)

[3.3. Tumor Size vs Age Range Grouped by Menopause Status 8](#_Toc179510378)

[3.4. Distribution of Degree of Malignancy 9](#_Toc179510379)

[3.5. Degree of Malignancy by Breast Quadrant 10](#_Toc179510380)

[3.6. Distribution of Age 10](#_Toc179510381)

[3.7. Scatter Plot of Tumor Size vs Age 12](#_Toc179510382)

[3.8. Distribution of Classes Across Age Ranges 12](#_Toc179510383)

[3.9. Histogram of All Numerical Features 13](#_Toc179510384)

[4. Machine Learning: 14](#_Toc179510385)

[4.1. Data Preparation 14](#_Toc179510386)

[4.2. Model Selection and Evaluation 14](#_Toc179510387)

[4.3. Model Comparison 15](#_Toc179510388)

[4.4. Best Performing Models 16](#_Toc179510389)

[5. Jupiter Notebook Codes 18](#_Toc179510390)

[Data Understanding: 18](#_Toc179510391)

[Gathering Data/Load dataset 18](#_Toc179510392)

[Describing data 19](#_Toc179510393)

[Exploring data and Verifying data quality 20](#_Toc179510394)

[Data Cleaning: 26](#_Toc179510395)

[Missing Values 26](#_Toc179510396)

[Data Redundancy 35](#_Toc179510397)

[Typos 37](#_Toc179510398)

[Outliers 38](#_Toc179510399)

[VISUALIZATION 38](#_Toc179510400)

[Machine Learning 47](#_Toc179510401)

[6. Conclusion: 51](#_Toc179510402)

# Group Member List:

|  |  |  |
| --- | --- | --- |
| **SN** | **Name** | **Student Number** |
| 1 | Orilio Naobeb | 223077593 |
| 2 | Jerome JEG Nanuseb | 223033030 |
| 3 | Beaven Muyoba | 223002291 |
| 4 | Loide Nambundunga | 221060154 |
| 5 | Nakaleke Fillemon | 223098523 |
| 6 | Emma Muulyao | 223066745 |
| 7 | Asser Naikuti | 223085065 |
| 8 | Lewis Mushukutu | 222129816 |
| 9 | Kamwi Mwilima | 223070068 |
| 10 | Kasparov Nangula | 222129182 |

# Introduction

### **Problem Statement:**

The Breast Cancer Dataset is collected to help develop machine learning models that can predict the likelihood of breast cancer recurrence in patients. Specifically, the dataset aims to address the following problem:

**Purpose:**

The primary goal is to build a predictive model that can determine whether a patient is likely to experience a recurrence of breast cancer after treatment. By analyzing patient characteristics such as age, tumor size, and lymph node involvement, the model can assist physicians in making informed decisions about further treatments, follow-up strategies, and patient care.

**Problem to Solve:**

The dataset is intended to help classify patients into two groups: those who will experience no recurrence of breast cancer and those who will experience a recurrence. Early and accurate predictions of recurrence are crucial for improving treatment outcomes and optimizing patient care.

**Why Use This Data:**

The data contains key medical information about patients, such as age, tumor size, and lymph node status, which are known factors influencing cancer recurrence. Machine learning models can be trained on this data to identify patterns that are not immediately obvious to human experts.

## 1. Column Understanding and Data Types:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Feature** | **Type** | **Values / Ranges** | **Description** | **Missing Values** |
| Class (Target) | Categorical (nominal) | no-recurrence-events, recurrence-events | Indicates whether the patient has experienced a recurrence of breast cancer or not. This is the target variable for the predictive model. |  |
| Age | Categorical (ordinal) | Ranges: 10-19, 20-29, 30-39, 40-49, 50-59, 60-69, 70-79, 80-89, 90-99 | The age of the patient in 10-year intervals. Age is often a significant factor in cancer prognosis. |  |
| Menopause | Categorical (nominal) | lt40, ge40, premeno | Indicates the patient’s menopausal status. Hormonal changes related to menopause can impact cancer risk and recurrence. |  |
| Tumor Size | Categorical (ordinal) | Ranges: 0-4, 5-9, 10-14, 15-19, 20-24, 25-29, 30-34, 35-39, 40-44, 45-49, 50-54, 55-59 | The size of the tumor in millimeters (mm). Larger tumors may have a higher risk of recurrence. |  |
| Inv-nodes | Categorical (ordinal) | Ranges: 0-2, 3-5, 6-8, 9-11, 12-14, 15-17, 18-20, 21-23, 24-26, 27-29, 30-32, 33-35, 36-39 | The number of involved lymph nodes. The higher the number, the greater the chance of cancer spreading. |  |
| Node-caps | Categorical (nominal) | yes, no | Indicates whether there is a capsular invasion (spread of cancer cells beyond the lymph node boundary). | 8 missing values |
| Deg-malig | Categorical (ordinal) | 1 (low), 2 (medium), 3 (high) | The degree of malignancy or severity of the cancer cells. A higher degree of malignancy indicates more aggressive cancer. |  |
| Breast | Categorical (nominal) | left, right | Indicates which breast was affected by cancer. This helps in analyzing if the recurrence is related to the location. |  |
| Breast-quad | Categorical (nominal) | left-up, left-low, right-up, right-low, central | Identifies the quadrant of the breast where the tumor was located. Different quadrants may have varying recurrence risks. | 1 missing value |
| Irradiat | Categorical (nominal) | yes, no | Indicates whether the patient received radiation therapy. Radiation is often used to reduce recurrence risk. |  |

## 2. Data Cleaning:

Within this section, we address missing values, duplicates, and potential outliers in our breast cancer dataset. The primary focus was on handling missing values in the 'Node-caps' and 'Breast-Quad' columns.

**Initial Data Inspection:**

Examining the last 60 rows of the dataset to get an initial view of the data. This allowed us to identify any obvious issues or patterns in the data.

* breast\_cancer.tail(60)

### 2.1. Handling Missing Values

2.1.1. 'Node-caps' Column

We separated the dataset into two parts: one with complete 'Node-caps' data and another with missing values.

For the missing values, we used a **Decision Tree Classifier** to predict the missing 'Node-caps' values:

* We prepared the training data by dropping 'Node-caps' and 'Breast-Quad' columns.
* The target variable ('Node-caps') was encoded using the LabelEncoder.
* Categorical features were one-hot encoded.
* A Decision Tree Classifier was trained on the complete data.
* Missing values were predicted using this trained model.
* Predicted values were then decoded back to original categories and used to fill the missing values.

2.1.2. 'Breast-Quad' Column

A similar process was followed for the 'Breast-Quad' column:

1. The dataset was split into complete and missing parts.

2. A Decision Tree Classifier was used to predict missing 'Breast-Quad' values.

3. The process of encoding, training, predicting, and updating was similar to that used for 'Node-caps'.

### 2.2. Verification of Missing Value Handling

After handling the missing values, we checked to ensure all missing values were addressed:

* breast\_cancer.isnull().sum()

This step above confirms that no null values remain in the dataset.

### 2.3. Final Data Inspection

We then examined the last 60 rows of the dataset again:

* breast\_cancer.tail(60)

This final check allows us to verify that the data cleaning process was successful and that the dataset is now complete.

2.4. Data Redundancy

## We handled data redundancy in the breast cancer dataset by first identifying all duplicate rows using:

## breast\_cancer.duplicated(keep=False)

## Which flagged every instance of duplication. These were displayed for review.

## drop\_duplicates(keep='first')

## Then removed the duplicates, keeping only the first occurrence.

## duplicated().sum().

## To confirmed no duplicates remained.

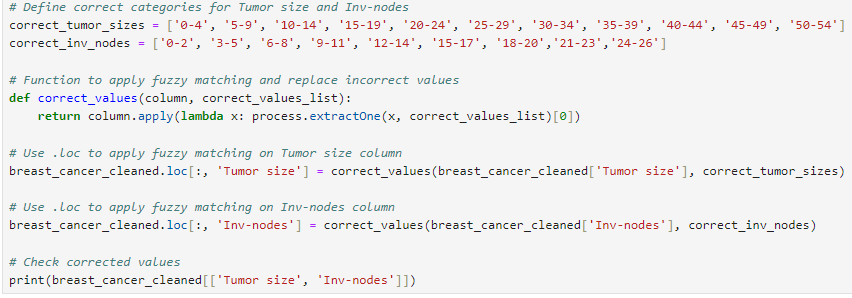
## This ensured the dataset was clean and ready for analysis.

2.5. Typos and Misspelling

We addressed typos in categorical columns by first reviewing the unique values in the 'Tumor size' and 'Inv-nodes' columns:

* print(breast\_cancer\_cleaned['Inv-nodes'].unique())
* print(breast\_cancer\_cleaned['Tumor size'].unique())

We then defined the correct categories for each column and applied fuzzy matching to correct any discrepancies:



This process ensured that the categories were consistent and corrected for any typos or inconsistencies in the dataset.

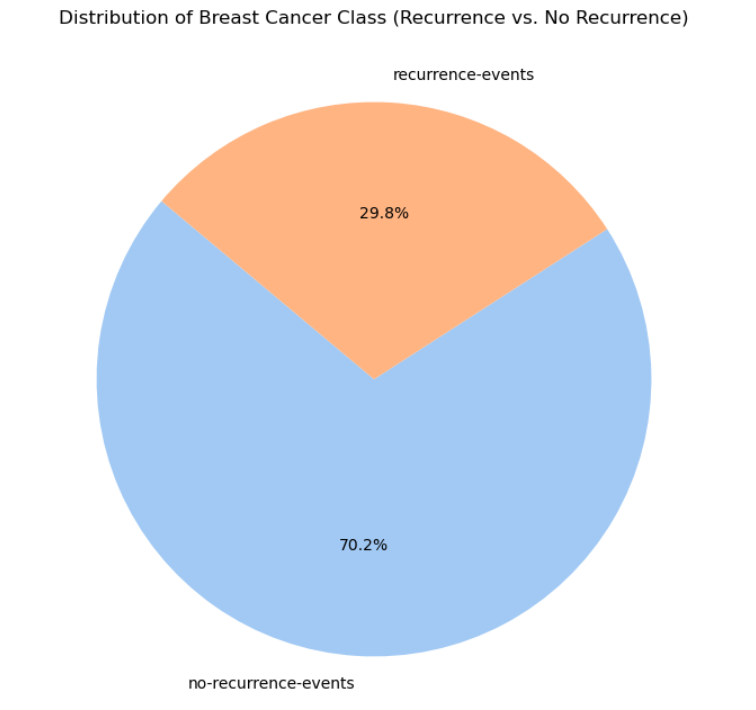
2.6. Outliers

In this project, outlier cleaning was unnecessary for the 'Degree of Malignancy' since it contains only three distinct values (1, 2, and 3), with no outliers present. Additionally, the remaining categorical data did not require outlier detection, as this process is only applicable to numerical data.

## 3. Exploratory Data:

### 3.1. Distribution of Breast Cancer Class

We examine the distribution of our target variable, the breast cancer class (recurrence vs no recurrence).



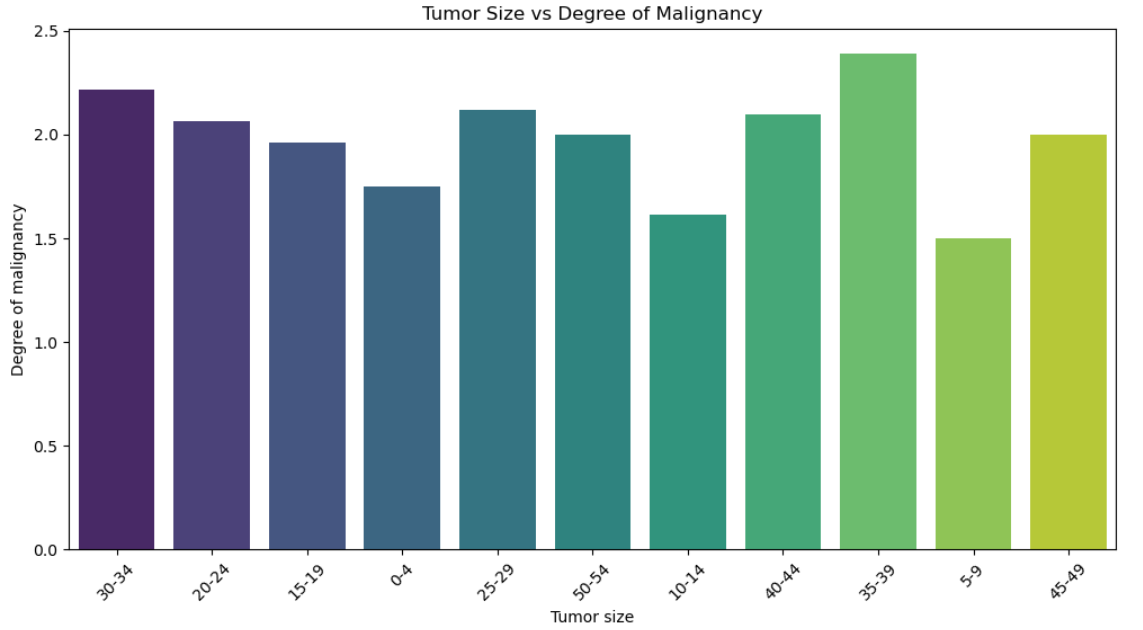
The pie chart reveals that:

* No-recurrence events: 70.6% of cases
* Recurrence events: 29.4% of cases

This distribution indicates an imbalance in our dataset, with a higher proportion of patients not experiencing cancer recurrence. This imbalance should be considered when interpreting model performance and potentially addressed through techniques like oversampling or under sampling.

### 3.2. Tumor Size vs. Degree of Malignancy

We then explored the relationship between tumor size and the degree of malignancy.

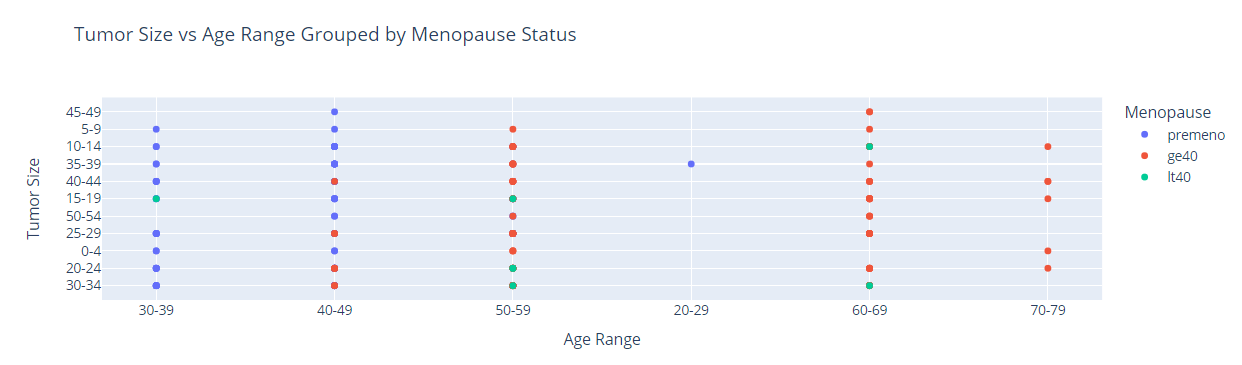


Observations from this bar plot:

* There appears to be a general trend where larger tumor sizes are associated with higher degrees of malignancy.
* However, this relationship is not strictly linear, as some smaller tumor sizes also show high degrees of malignancy.
* The variability in degree of malignancy seems to increase with tumor size, suggesting that other factors may influence malignancy for larger tumors.

### 3.3. Tumor Size vs Age Range Grouped by Menopause Status

We used an interactive scatter plot to visualize the relationship between tumor size, age range, and menopause status.



Observations from the scatter plot:

* Tumor sizes vary across all age ranges, indicating that age alone is not a determining factor for tumor size.
* The 'premeno' group appears to have a wider range of tumor sizes compared to other menopause statuses.
* There seems to be a slight trend of larger tumors in older age groups, but this is not consistent across all data points.

### 3.4. Distribution of Degree of Malignancy

We examine the distribution of the degree of malignancy to understand the severity of cases in our dataset.

A graph with a line graph

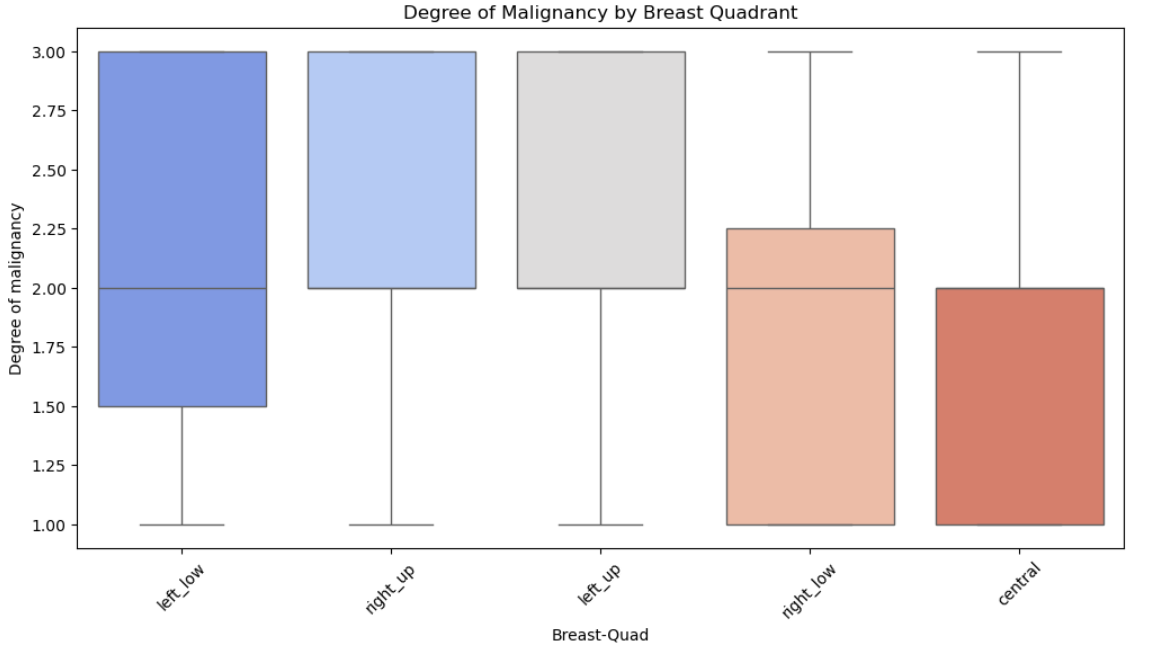
Description automatically generated with medium confidence

Observations from the histogram:

* The distribution of malignancy degrees is not normal, showing a skew towards lower degrees.
* The most common degree of malignancy is 2, followed by 3, with 1 being the least common.
* This distribution suggests that moderate to high-grade tumors are more prevalent in the dataset.

### 3.5. Degree of Malignancy by Breast Quadrant

We investigate how the degree of malignancy varies across different breast quadrants.

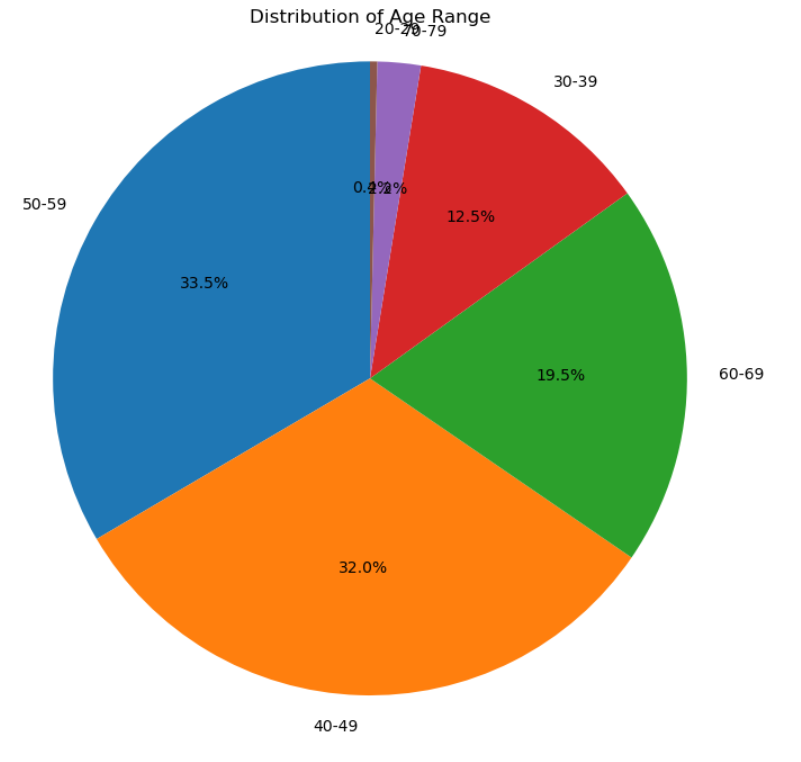


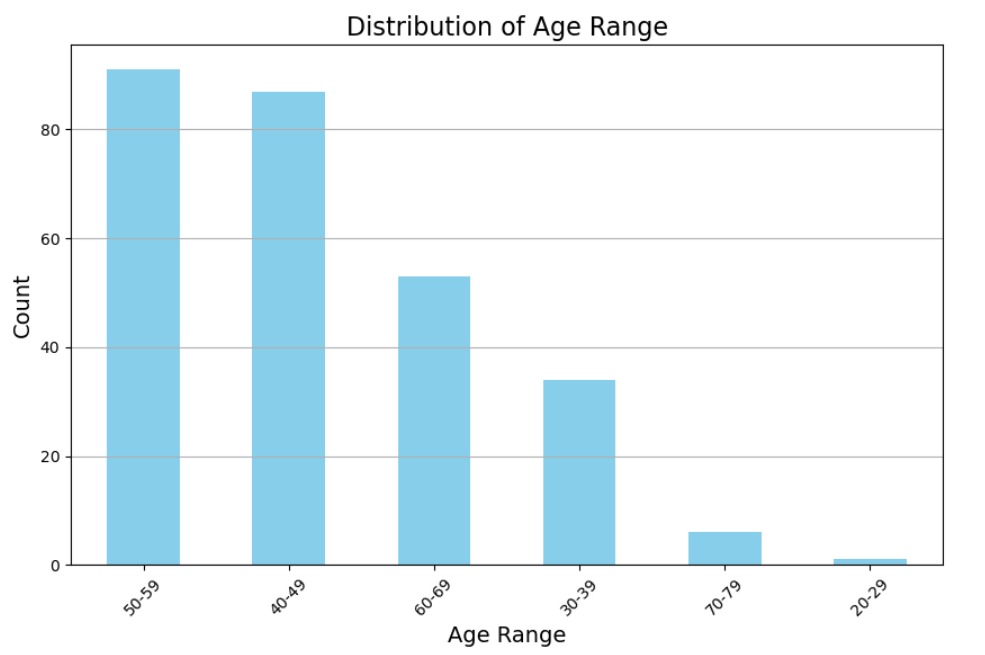
Observations from the box plot:

* The central quadrant shows the highest median degree of malignancy, suggesting that tumors in this area may be more aggressive.
* The left-low quadrant has the widest range of malignancy degrees, indicating high variability in tumor aggressiveness in this area.
* Right-up and right-low quadrants show similar distributions, with slightly lower median malignancy compared to other quadrants.

### 3.6. Distribution of Age

We investigate the distribution of patient ages using a pie chart and a histogram.



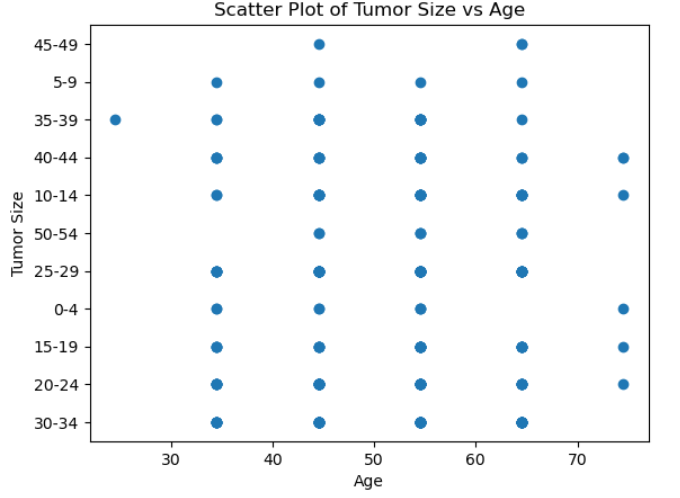


Observations from the age distribution charts:

* The 50-59 age group appears to be the largest segment, followed by 40-49.
* There are fewer cases in the youngest and oldest age ranges, which may affect the model's ability to predict outcomes for these groups.

### 3.7. Scatter Plot of Tumor Size vs Age

We examine the relationship between tumor size and patient age using a scatter plot.

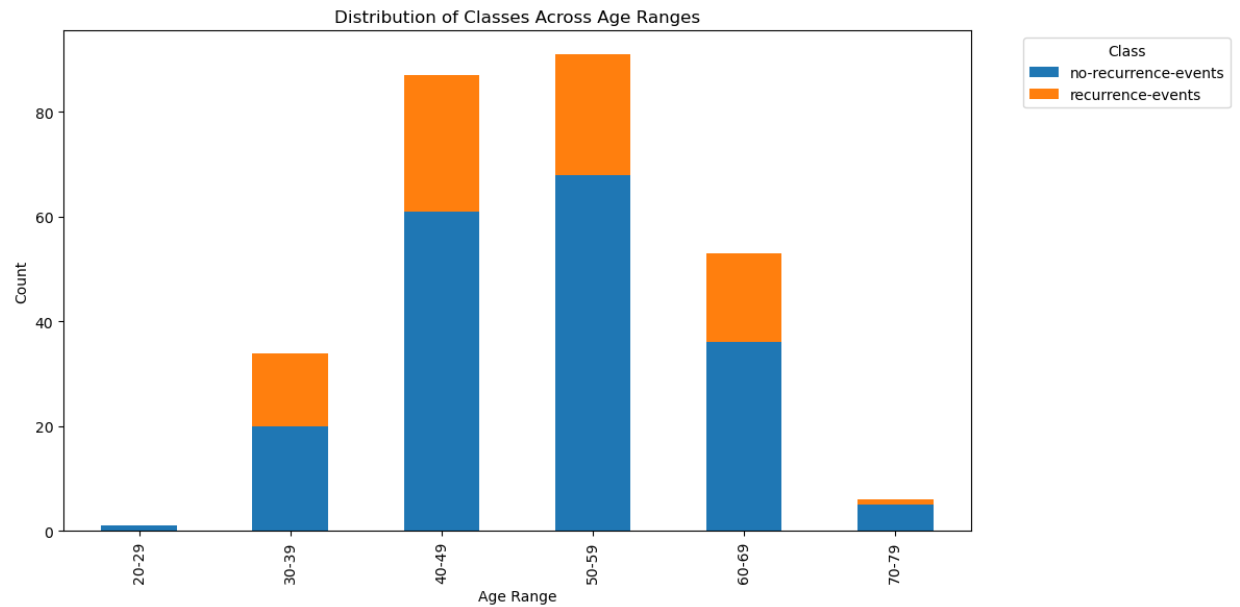


Observations from the scatter plot:

* There doesn't appear to be a strong correlation between age and tumor size.
* Tumor sizes vary widely across all age groups, suggesting that other factors may be more influential in determining tumor size.

### 3.8. Distribution of Classes Across Age Ranges

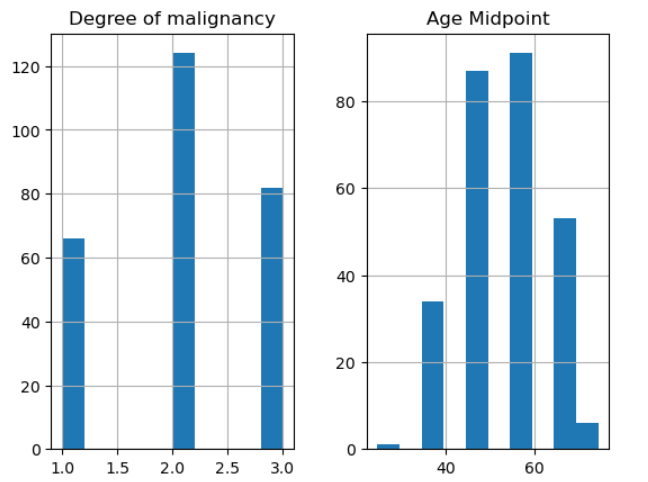
We analyze how the distribution of recurrence and non-recurrence cases varies across different age ranges.



Observations from the stacked bar chart:

* The proportion of recurrence cases seems to vary across age ranges.
* Some age groups may have a higher recurrence rate, which could be an important factor for the predictive model.

### 3.9. Histogram of All Numerical Features



4. Machine Learning:

### 4.1. Data Preparation

Before applying machine learning algorithms, we need to prepare our data. This involves encoding categorical variables and splitting the dataset into training and validation sets.



### 4.2. Model Selection and Evaluation

We evaluate several machine learning algorithms to determine which performs best for our breast cancer recurrence prediction task.



Results of model evaluation:

* LR: 0.774162 (0.085047)
* LDA: 0.773669 (0.081565)

### KNN: 0.693294 (0.102990)

### CART: 0.677022 (0.134593)

### NB: 0.758383 (0.119856)

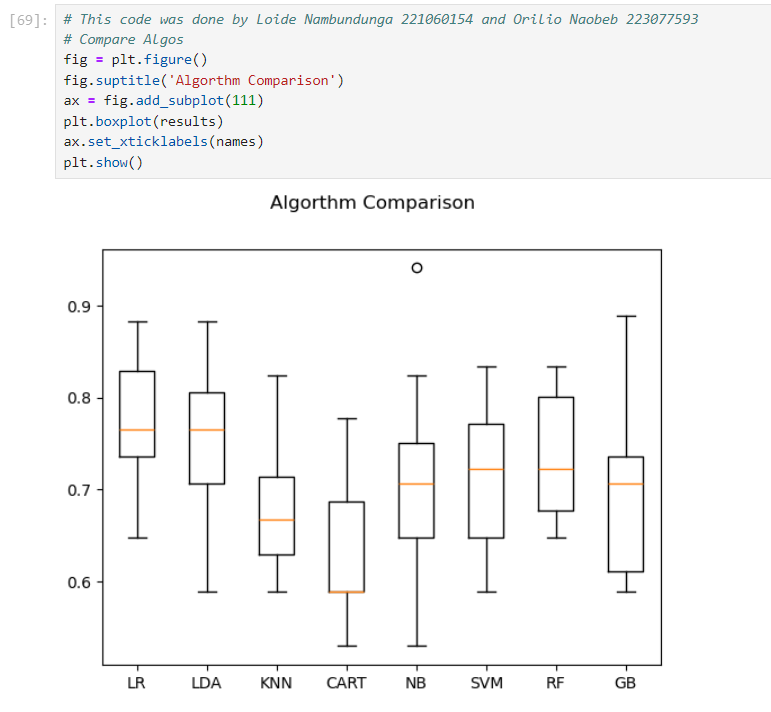
### SVM: 0.748028 (0.107158)

### RF: 0.760848 (0.075571)

### GB: 0.724852 (0.108333)

### 4.3. Model Comparison

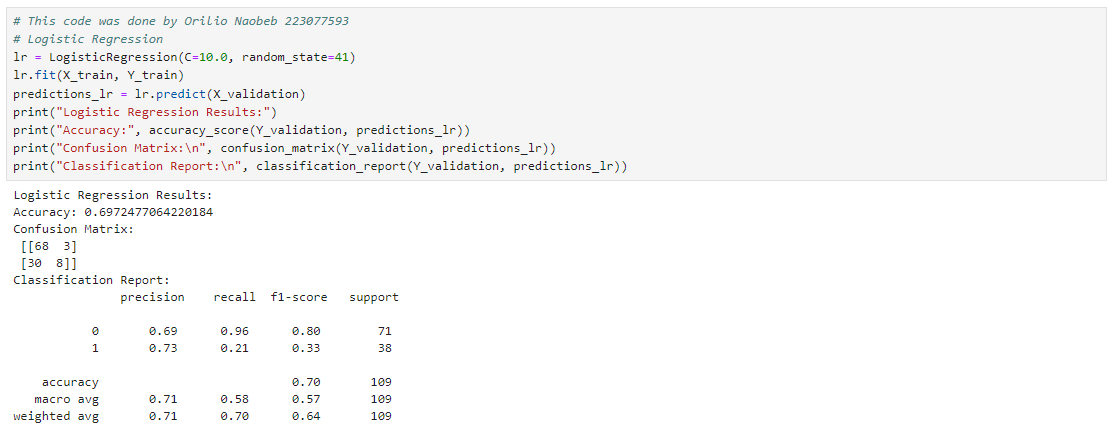
We visualize the performance of different models using a box plot. The box plot allows us to compare the distribution of accuracy scores for each algorithm, helping us identify the most promising models for our project.



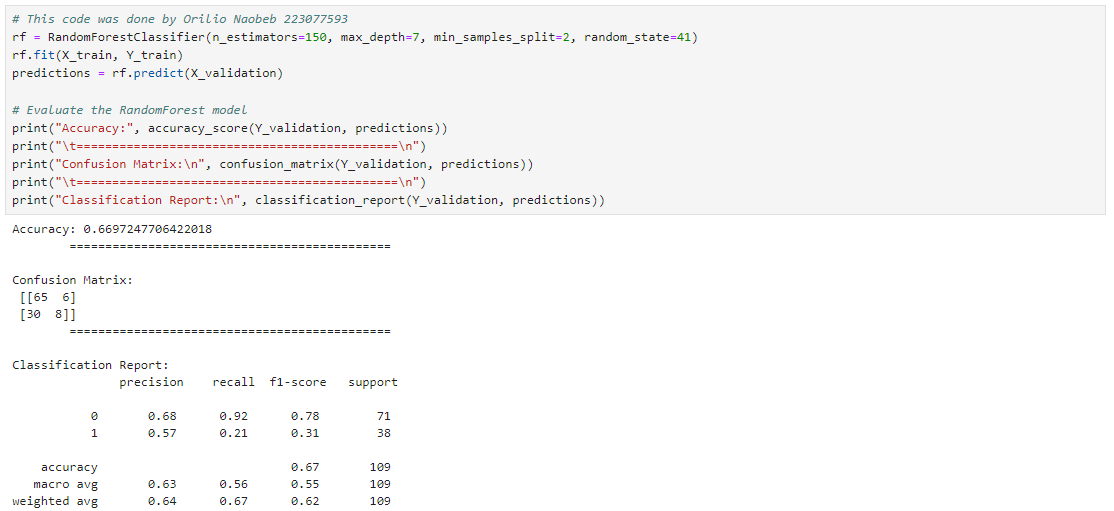
### 4.4. Best Performing Models

Based on the evaluation results, we select the top-performing models for further analysis: Logistic Regression, Random Forest, and Linear Discriminant Analysis.

4.4.1. Logistic Regression



4.4.2. Random Forest

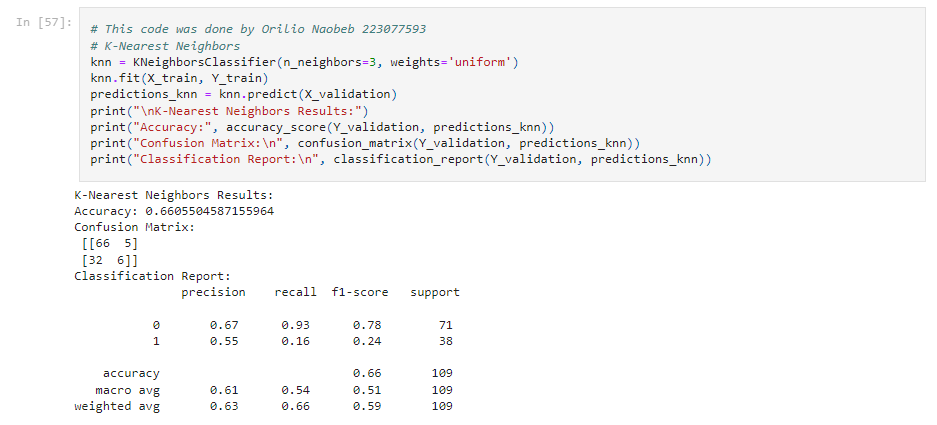


4.4.3. Linear Discriminant Analysis.

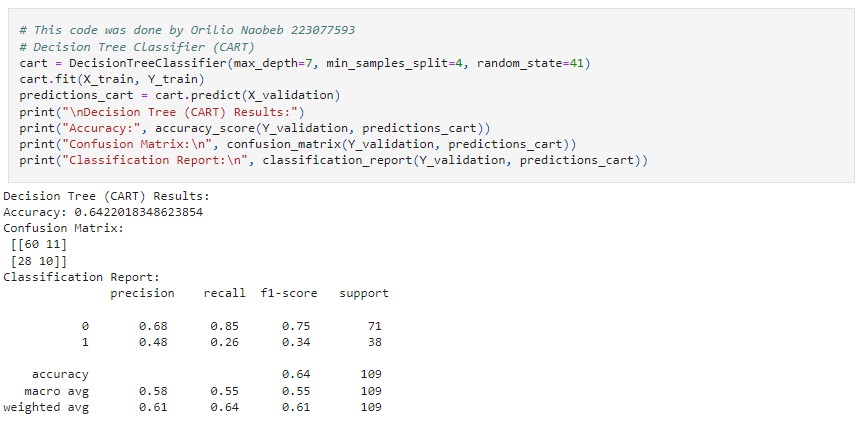
A screenshot of a computer program

Description automatically generated

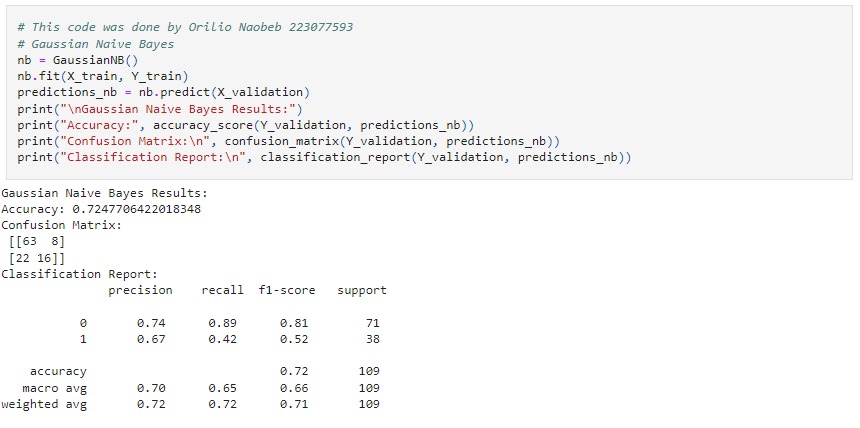
4.4.4. K-Nearest Neighbors Classifier



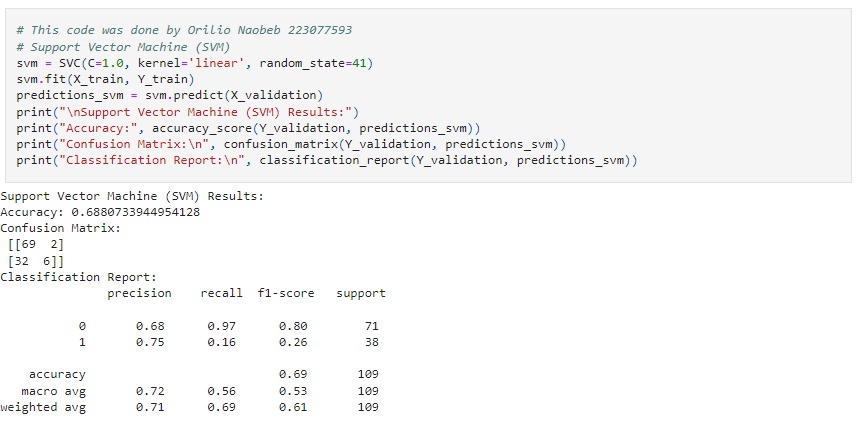
4.4.5. Decision Tree Classifier

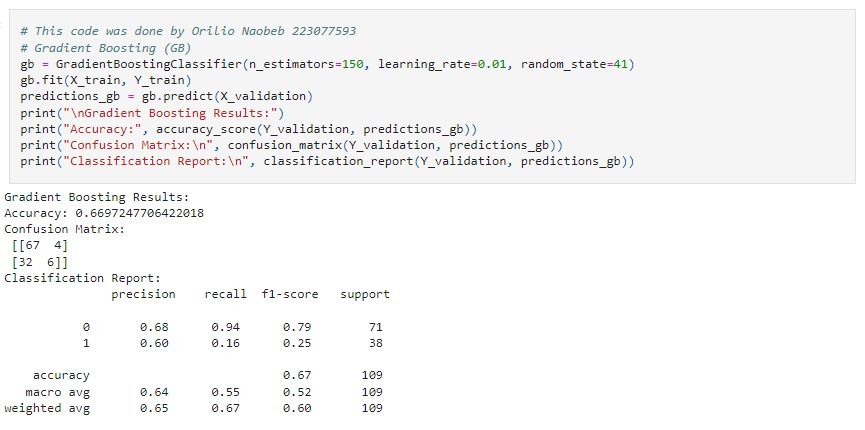


4.4.6. Gaussian Naive Bayes



4.4.7. Support Vector Machine

4.4.8. Gradient Boosting Classifier



4.5. Choosing best model

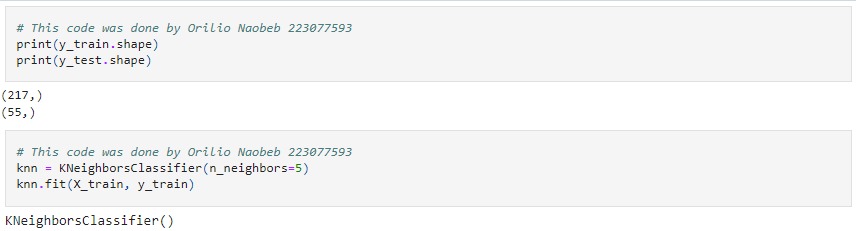
* The LabelEncoder is used to convert categorical labels into numeric form. In this case, it's encoding the target variable, Class, which likely indicates whether a breast cancer case is malignant or benign.



A white rectangular object with a black border

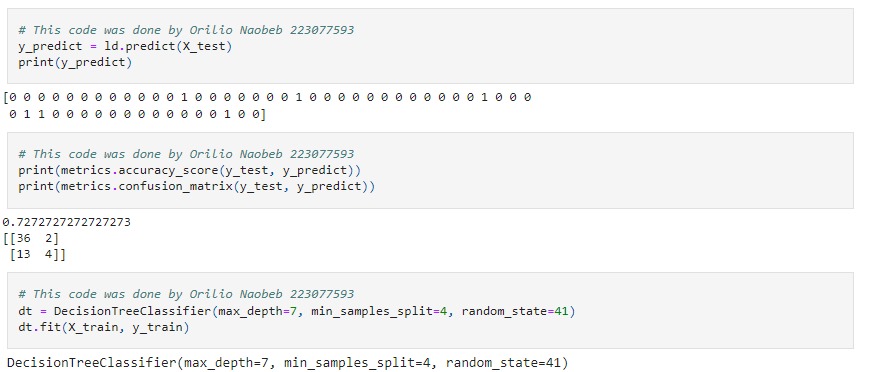
Description automatically generated with medium confidence









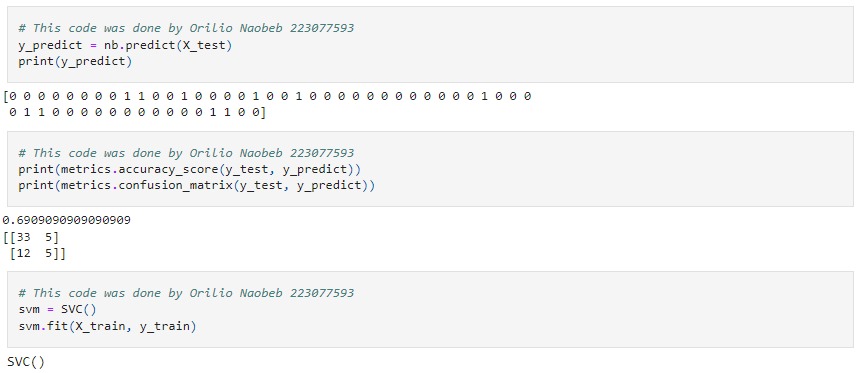


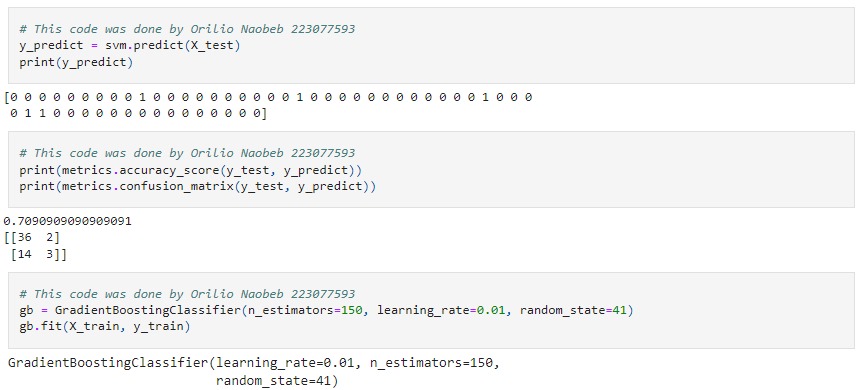
A screenshot of a computer

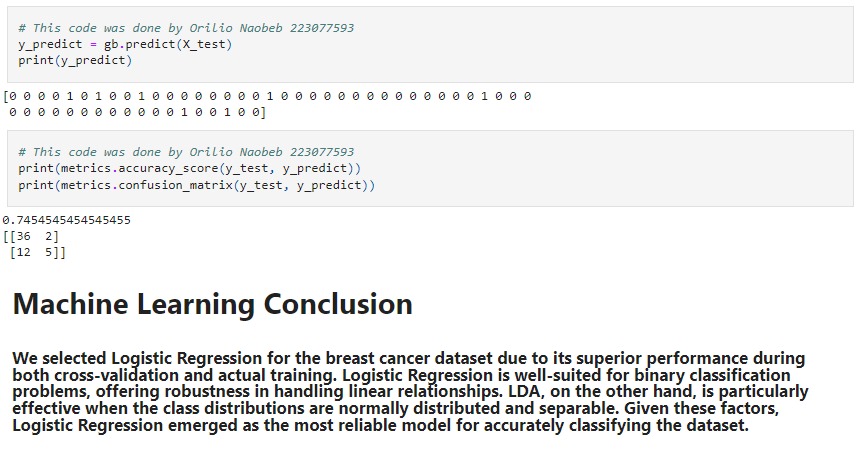
Description automatically generated

A screenshot of a computer

Description automatically generated







5. Jupiter Notebook Codes

Imports

# This code was done by 223077593 - ORILIO NAOBEB  
import pandas as pd  
import numpy as np  
  
from sklearn.impute import SimpleImputer  
from sklearn.preprocessing import StandardScaler  
from sklearn.preprocessing import LabelEncoder  
from sklearn.tree import DecisionTreeRegressor  
from sklearn.tree import DecisionTreeClassifier  
from thefuzz import process  
  
import matplotlib.pyplot as plt  
import seaborn as sns  
  
# This code was done by 222129816 - TANAKA MUSHUKUTU  
import plotly.express as px  
#223085065 Asser Naikuti  
from scipy.stats import norm  
  
from sklearn import model\_selection  
from sklearn.metrics import classification\_report  
from sklearn.metrics import confusion\_matrix  
from sklearn.metrics import accuracy\_score  
  
from sklearn.linear\_model import LogisticRegression  
from sklearn.tree import DecisionTreeClassifier  
from sklearn.neighbors import KNeighborsClassifier  
from sklearn.discriminant\_analysis import LinearDiscriminantAnalysis  
from sklearn.naive\_bayes import GaussianNB  
from sklearn.svm import SVC  
from sklearn.ensemble import RandomForestClassifier, GradientBoostingClassifier  
  
from sklearn.model\_selection import train\_test\_split  
from sklearn import metrics

# **Data Understanding:**

## **Gathering Data/Load dataset**

# This code was done by 223077593 - ORILIO NAOBEB  
breast\_cancer= pd.read\_csv("breast-cancer.csv")

# This code was done by 223077593 - ORILIO NAOBEB  
breast\_cancer

Age Range Menopause Tumor size Inv-nodes Node-caps Degree of malignancy \  
0 30-39 premeno 30-34 0-2 no 3   
1 40-49 premeno 20-24 0-2 no 2   
2 40-49 premeno 20-24 0-2 no 2   
3 60-69 ge40 15-19 0-2 no 2   
4 40-49 premeno 0-4 0-2 no 2   
.. ... ... ... ... ... ...   
281 30-39 premeno 30-34 0-2 no 2   
282 30-39 premeno 20-24 0-2 no 3   
283 60-69 ge40 20-24 0-2 no 1   
284 40-49 ge40 30-34 5-Mar no 3   
285 50-59 ge40 30-34 5-Mar no 3   
  
 Breast Breast-Quad Irradiat Class   
0 left left\_low no no-recurrence-events   
1 right right\_up no no-recurrence-events   
2 left left\_low no no-recurrence-events   
3 right left\_up no no-recurrence-events   
4 right right\_low no no-recurrence-events   
.. ... ... ... ...   
281 left left\_up no recurrence-events   
282 left left\_up yes recurrence-events   
283 right left\_up no recurrence-events   
284 left left\_low no recurrence-events   
285 left left\_low no recurrence-events   
  
[286 rows x 10 columns]

## **Describing data**

# This code was done by 223077593 - ORILIO NAOBEB  
breast\_cancer.info()

<class 'pandas.core.frame.DataFrame'>  
RangeIndex: 286 entries, 0 to 285  
Data columns (total 10 columns):  
 # Column Non-Null Count Dtype   
--- ------ -------------- -----   
 0 Age Range 286 non-null object  
 1 Menopause 286 non-null object  
 2 Tumor size 286 non-null object  
 3 Inv-nodes 286 non-null object  
 4 Node-caps 278 non-null object  
 5 Degree of malignancy 286 non-null int64   
 6 Breast 286 non-null object  
 7 Breast-Quad 285 non-null object  
 8 Irradiat 286 non-null object  
 9 Class 286 non-null object  
dtypes: int64(1), object(9)  
memory usage: 22.5+ KB

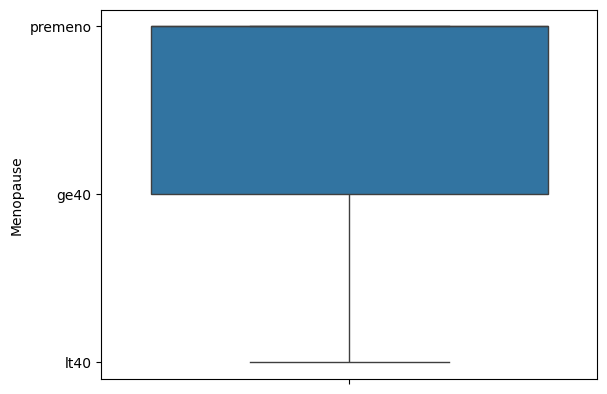
# This code was done by 223077593 - ORILIO NAOBEB  
breast\_cancer.describe()

Degree of malignancy  
count 286.000000  
mean 2.048951  
std 0.738217  
min 1.000000  
25% 2.000000  
50% 2.000000  
75% 3.000000  
max 3.000000

## **Exploring data and Verifying data quality**

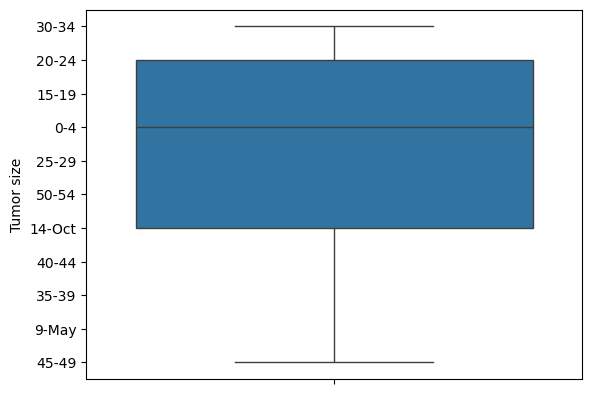
# This code was done by 223077593 - ORILIO NAOBEB  
sns.boxplot(breast\_cancer['Menopause'])

<Axes: ylabel='Menopause'>



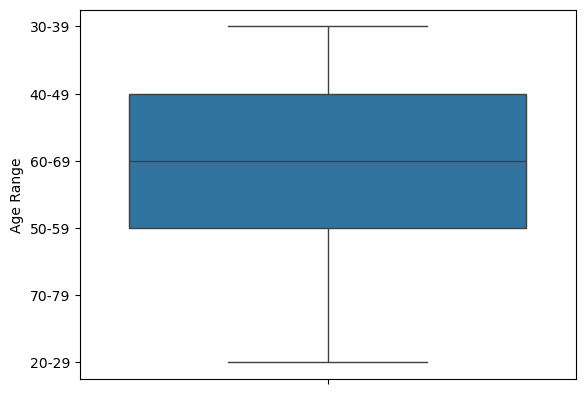
# This code was done by 223077593 - ORILIO NAOBEB  
sns.boxplot(breast\_cancer['Tumor size'])

<Axes: ylabel='Tumor size'>



# This code was done by 223077593 - ORILIO NAOBEB  
sns.boxplot(breast\_cancer['Age Range'])

<Axes: ylabel='Age Range'>



# This code was done by 223077593 - ORILIO NAOBEB  
sns.boxplot(breast\_cancer['Inv-nodes'])

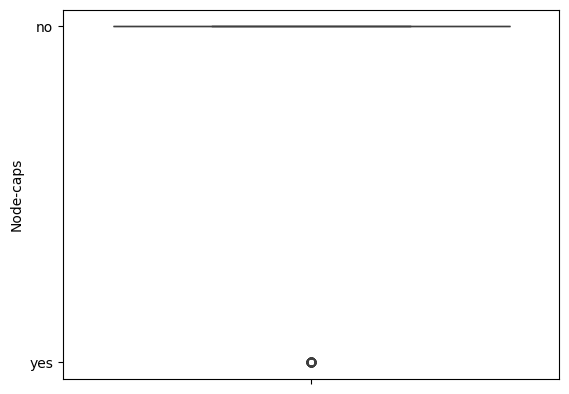
<Axes: ylabel='Inv-nodes'>

A blue rectangular object with black dots

Description automatically generated

# This code was done by 223077593 - ORILIO NAOBEB  
sns.boxplot(breast\_cancer['Node-caps'])

<Axes: ylabel='Node-caps'>



# This code was done by 223077593 - ORILIO NAOBEB  
sns.boxplot(breast\_cancer['Degree of malignancy'])

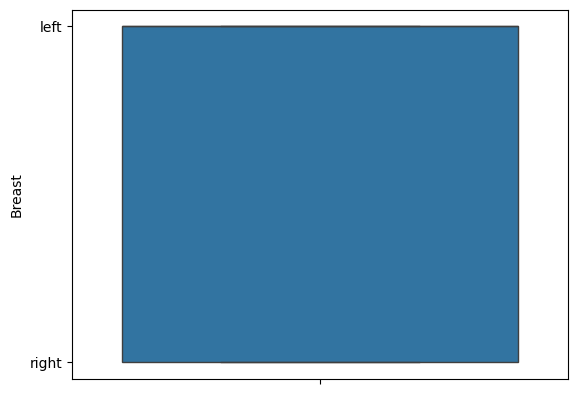
<Axes: ylabel='Degree of malignancy'>

A blue rectangular object with black lines

Description automatically generated

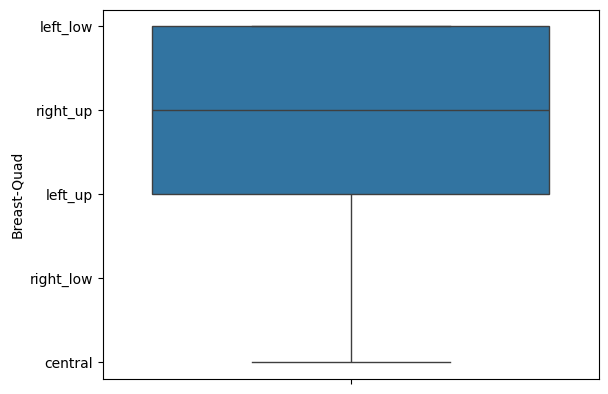
# This code was done by 223077593 - ORILIO NAOBEB  
sns.boxplot(breast\_cancer['Breast'])

<Axes: ylabel='Breast'>



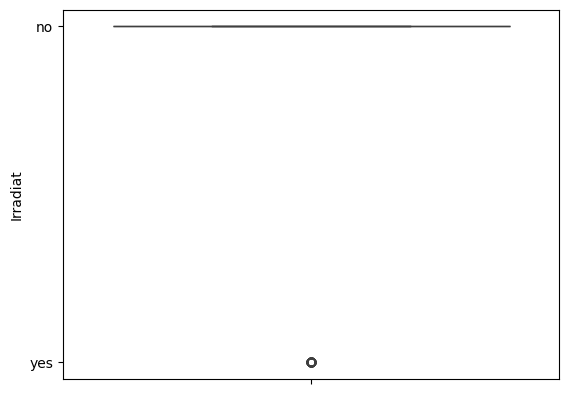
# This code was done by 223077593 - ORILIO NAOBEB  
sns.boxplot(breast\_cancer['Breast-Quad'])

<Axes: ylabel='Breast-Quad'>



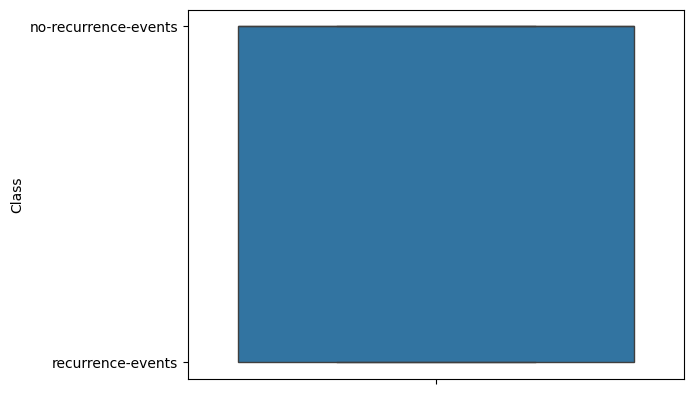
# This code was done by 223077593 - ORILIO NAOBEB  
sns.boxplot(breast\_cancer['Irradiat'])

<Axes: ylabel='Irradiat'>



# This code was done by 223077593 - ORILIO NAOBEB  
sns.boxplot(breast\_cancer['Class'])

<Axes: ylabel='Class'>



# This code was done by 223077593 - ORILIO NAOBEB  
print(breast\_cancer['Menopause'].unique())

['premeno' 'ge40' 'lt40']

# This code was done by 223077593 - ORILIO NAOBEB  
print(breast\_cancer['Node-caps'].unique())

['no' 'yes' nan]

# This code was done by 223077593 - ORILIO NAOBEB  
print(breast\_cancer['Breast'].unique())

['left' 'right']

# This code was done by 223077593 - ORILIO NAOBEB  
print(breast\_cancer['Breast-Quad'].unique())

['left\_low' 'right\_up' 'left\_up' 'right\_low' 'central' nan]

# This code was done by 223077593 - ORILIO NAOBEB  
print(breast\_cancer['Irradiat'].unique())

['no' 'yes']

# This code was done by 223077593 - ORILIO NAOBEB  
print(breast\_cancer['Inv-nodes'].unique())

['0-2' '8-Jun' '11-Sep' '5-Mar' '15-17' '14-Dec' '24-26']

# This code was done by 223077593 - ORILIO NAOBEB  
print(breast\_cancer['Tumor size'].unique())

['30-34' '20-24' '15-19' '0-4' '25-29' '50-54' '14-Oct' '40-44' '35-39'  
 '9-May' '45-49']

# This code was done by 223077593 - ORILIO NAOBEB  
print(breast\_cancer['Age Range'].unique())

['30-39' '40-49' '60-69' '50-59' '70-79' '20-29']

# This code was done by 223077593 - ORILIO NAOBEB  
print(breast\_cancer['Degree of malignancy'].unique())

[3 2 1]

# This code was done by 223077593 - ORILIO NAOBEB  
print(breast\_cancer['Class'].unique())

['no-recurrence-events' 'recurrence-events']

# This code was done by 223077593 - ORILIO NAOBEB  
breast\_cancer.isnull().sum()

Age Range 0  
Menopause 0  
Tumor size 0  
Inv-nodes 0  
Node-caps 8  
Degree of malignancy 0  
Breast 0  
Breast-Quad 1  
Irradiat 0  
Class 0  
dtype: int64

# This code was done by 223077593 - ORILIO NAOBEB  
breast\_cancer.duplicated().sum()

14

# **Data Cleaning:**

## **Missing Values**

# This code was done by 223077593 - ORILIO NAOBEB  
breast\_cancer.tail(60)

Age Range Menopause Tumor size Inv-nodes Node-caps Degree of malignancy \  
226 40-49 premeno 30-34 15-17 yes 3   
227 50-59 premeno 30-34 0-2 no 3   
228 60-69 ge40 40-44 5-Mar yes 3   
229 60-69 ge40 45-49 0-2 no 1   
230 50-59 premeno 50-54 11-Sep yes 2   
231 40-49 premeno 30-34 5-Mar no 2   
232 30-39 premeno 30-34 5-Mar no 3   
233 70-79 ge40 15-19 11-Sep NaN 1   
234 60-69 ge40 30-34 0-2 no 3   
235 50-59 premeno 25-29 5-Mar yes 3   
236 40-49 premeno 25-29 0-2 no 2   
237 40-49 premeno 25-29 0-2 no 2   
238 30-39 premeno 35-39 0-2 no 3   
239 40-49 premeno 20-24 5-Mar yes 2   
240 60-69 ge40 20-24 5-Mar no 2   
241 40-49 premeno 15-19 15-17 yes 3   
242 50-59 ge40 25-29 8-Jun no 3   
243 50-59 ge40 20-24 5-Mar yes 3   
244 40-49 premeno 30-34 14-Dec yes 3   
245 30-39 premeno 30-34 11-Sep no 2   
246 30-39 premeno 15-19 8-Jun yes 3   
247 50-59 ge40 30-34 11-Sep yes 3   
248 60-69 ge40 35-39 8-Jun yes 3   
249 30-39 premeno 20-24 5-Mar yes 2   
250 40-49 premeno 25-29 0-2 no 3   
251 40-49 premeno 50-54 0-2 no 2   
252 30-39 premeno 40-44 0-2 no 1   
253 60-69 ge40 50-54 0-2 no 3   
254 40-49 premeno 30-34 0-2 yes 3   
255 40-49 premeno 30-34 8-Jun yes 3   
256 40-49 premeno 30-34 0-2 no 1   
257 40-49 premeno 20-24 5-Mar yes 2   
258 50-59 ge40 30-34 8-Jun yes 2   
259 50-59 ge40 30-34 5-Mar no 3   
260 60-69 ge40 25-29 5-Mar no 2   
261 40-49 ge40 25-29 14-Dec yes 3   
262 60-69 ge40 25-29 0-2 no 3   
263 50-59 lt40 20-24 0-2 NaN 1   
264 50-59 lt40 20-24 0-2 NaN 1   
265 30-39 premeno 35-39 11-Sep yes 3   
266 40-49 premeno 30-34 5-Mar yes 2   
267 60-69 ge40 20-24 24-26 yes 3   
268 30-39 premeno 35-39 0-2 no 3   
269 40-49 premeno 25-29 0-2 no 2   
270 50-59 ge40 30-34 8-Jun yes 3   
271 50-59 premeno 25-29 0-2 no 3   
272 40-49 premeno 15-19 0-2 yes 3   
273 60-69 ge40 30-34 0-2 yes 2   
274 60-69 ge40 30-34 5-Mar yes 2   
275 40-49 premeno 25-29 11-Sep yes 3   
276 30-39 premeno 25-29 8-Jun yes 3   
277 60-69 ge40 14-Oct 8-Jun yes 3   
278 50-59 premeno 35-39 15-17 yes 3   
279 50-59 ge40 40-44 8-Jun yes 3   
280 50-59 ge40 40-44 8-Jun yes 3   
281 30-39 premeno 30-34 0-2 no 2   
282 30-39 premeno 20-24 0-2 no 3   
283 60-69 ge40 20-24 0-2 no 1   
284 40-49 ge40 30-34 5-Mar no 3   
285 50-59 ge40 30-34 5-Mar no 3   
  
 Breast Breast-Quad Irradiat Class   
226 left left\_low no recurrence-events   
227 right left\_up yes recurrence-events   
228 right left\_low no recurrence-events   
229 right right\_up yes recurrence-events   
230 right left\_up no recurrence-events   
231 right left\_up no recurrence-events   
232 right left\_up yes recurrence-events   
233 left left\_low yes recurrence-events   
234 right left\_up yes recurrence-events   
235 left left\_low yes recurrence-events   
236 right left\_low no recurrence-events   
237 right left\_low no recurrence-events   
238 left left\_low no recurrence-events   
239 right right\_up yes recurrence-events   
240 left left\_low yes recurrence-events   
241 left left\_low no recurrence-events   
242 left left\_low yes recurrence-events   
243 right right\_up no recurrence-events   
244 left left\_up yes recurrence-events   
245 right left\_up yes recurrence-events   
246 left left\_low yes recurrence-events   
247 left right\_low yes recurrence-events   
248 left left\_low no recurrence-events   
249 left left\_low no recurrence-events   
250 left left\_up no recurrence-events   
251 right left\_low yes recurrence-events   
252 left left\_up no recurrence-events   
253 right left\_up no recurrence-events   
254 right right\_up no recurrence-events   
255 right left\_up no recurrence-events   
256 left left\_low yes recurrence-events   
257 left left\_low yes recurrence-events   
258 left right\_low yes recurrence-events   
259 right left\_up no recurrence-events   
260 right right\_up no recurrence-events   
261 left right\_low yes recurrence-events   
262 left left\_up no recurrence-events   
263 left left\_up no recurrence-events   
264 left left\_low no recurrence-events   
265 left left\_low no recurrence-events   
266 left right\_up no recurrence-events   
267 left left\_low yes recurrence-events   
268 left left\_low no recurrence-events   
269 left left\_low yes recurrence-events   
270 left right\_low no recurrence-events   
271 right left\_low yes recurrence-events   
272 right left\_up no recurrence-events   
273 right right\_up yes recurrence-events   
274 left central yes recurrence-events   
275 right left\_up no recurrence-events   
276 left right\_low yes recurrence-events   
277 left left\_up yes recurrence-events   
278 right right\_up no recurrence-events   
279 left left\_low yes recurrence-events   
280 left left\_low yes recurrence-events   
281 left left\_up no recurrence-events   
282 left left\_up yes recurrence-events   
283 right left\_up no recurrence-events   
284 left left\_low no recurrence-events   
285 left left\_low no recurrence-events

# This code was done by 223077593 - ORILIO NAOBEB  
#Handle missing values for 'Node-caps'  
node\_caps\_complete = breast\_cancer[breast\_cancer['Node-caps'].notna()]  
node\_caps\_missing = breast\_cancer[breast\_cancer['Node-caps'].isna()]  
  
if not node\_caps\_missing.empty:  
 # Prepare training data  
 X\_train\_node\_caps = node\_caps\_complete.drop(['Node-caps', 'Breast-Quad'], axis=1)  
 y\_train\_node\_caps = node\_caps\_complete['Node-caps']  
  
 # Encode target variable  
 label\_encoder\_node\_caps = LabelEncoder()  
 y\_train\_node\_caps = label\_encoder\_node\_caps.fit\_transform(y\_train\_node\_caps)  
  
 # One-Hot Encoding for categorical features  
 X\_train\_node\_caps = pd.get\_dummies(X\_train\_node\_caps)  
  
 # Decision Tree for 'Node-caps' (using classifier)  
 tree\_clf\_node\_caps = DecisionTreeClassifier()  
 tree\_clf\_node\_caps.fit(X\_train\_node\_caps, y\_train\_node\_caps)  
  
 # Prepare features DataFrame for missing 'Node-caps'  
 X\_missing\_node\_caps = node\_caps\_missing.drop(['Node-caps', 'Breast-Quad'], axis=1)  
  
 # One-Hot Encoding for missing features  
 X\_missing\_node\_caps = pd.get\_dummies(X\_missing\_node\_caps)  
  
 # Ensure the columns match between training and missing data  
 X\_missing\_node\_caps = X\_missing\_node\_caps.reindex(columns=X\_train\_node\_caps.columns, fill\_value=0)  
  
 # Predict missing 'Node-caps' values  
 if not X\_missing\_node\_caps.empty:  
 predicted\_values\_node\_caps = tree\_clf\_node\_caps.predict(X\_missing\_node\_caps)  
 # Decode predicted values back to original categories  
 predicted\_values\_node\_caps = label\_encoder\_node\_caps.inverse\_transform(predicted\_values\_node\_caps)  
 # Update the missing 'Node-caps' values in the original DataFrame  
 breast\_cancer.loc[breast\_cancer['Node-caps'].isna(), 'Node-caps'] = predicted\_values\_node\_caps  
  
# Step 2: Handle 'Breast-Quad' in a similar way  
breast\_quad\_complete = breast\_cancer[breast\_cancer['Breast-Quad'].notna()]  
breast\_quad\_missing = breast\_cancer[breast\_cancer['Breast-Quad'].isna()]  
  
if not breast\_quad\_missing.empty:  
 X\_train\_breast\_quad = breast\_quad\_complete.drop(['Breast-Quad', 'Node-caps'], axis=1)  
 y\_train\_breast\_quad = breast\_quad\_complete['Breast-Quad']  
  
 # Encode target variable  
 label\_encoder\_breast\_quad = LabelEncoder()  
 y\_train\_breast\_quad = label\_encoder\_breast\_quad.fit\_transform(y\_train\_breast\_quad)  
  
 # One-Hot Encoding for categorical features  
 X\_train\_breast\_quad = pd.get\_dummies(X\_train\_breast\_quad)  
  
 # Decision Tree for 'Breast-Quad' (using classifier)  
 tree\_clf\_breast\_quad = DecisionTreeClassifier()  
 tree\_clf\_breast\_quad.fit(X\_train\_breast\_quad, y\_train\_breast\_quad)  
  
 # Prepare features DataFrame for missing 'Breast-Quad'  
 X\_missing\_breast\_quad = breast\_quad\_missing.drop(['Breast-Quad', 'Node-caps'], axis=1)  
  
 # One-Hot Encoding for missing features  
 X\_missing\_breast\_quad = pd.get\_dummies(X\_missing\_breast\_quad)  
  
 # Ensure the columns match between training and missing data  
 X\_missing\_breast\_quad = X\_missing\_breast\_quad.reindex(columns=X\_train\_breast\_quad.columns, fill\_value=0)  
  
 # Predict missing 'Breast-Quad' values  
 if not X\_missing\_breast\_quad.empty:  
 predicted\_values\_breast\_quad = tree\_clf\_breast\_quad.predict(X\_missing\_breast\_quad)  
 # Decode predicted values back to original categories  
 predicted\_values\_breast\_quad = label\_encoder\_breast\_quad.inverse\_transform(predicted\_values\_breast\_quad)  
 # Update the missing 'Breast-Quad' values in the original DataFrame  
 breast\_cancer.loc[breast\_cancer['Breast-Quad'].isna(), 'Breast-Quad'] = predicted\_values\_breast\_quad

# This code was done by 223077593 - ORILIO NAOBEB  
breast\_cancer.isnull().sum()

Age Range 0  
Menopause 0  
Tumor size 0  
Inv-nodes 0  
Node-caps 0  
Degree of malignancy 0  
Breast 0  
Breast-Quad 0  
Irradiat 0  
Class 0  
dtype: int64

# This code was done by 223077593 - ORILIO NAOBEB  
breast\_cancer.tail(60)

Age Range Menopause Tumor size Inv-nodes Node-caps Degree of malignancy \  
226 40-49 premeno 30-34 15-17 yes 3   
227 50-59 premeno 30-34 0-2 no 3   
228 60-69 ge40 40-44 5-Mar yes 3   
229 60-69 ge40 45-49 0-2 no 1   
230 50-59 premeno 50-54 11-Sep yes 2   
231 40-49 premeno 30-34 5-Mar no 2   
232 30-39 premeno 30-34 5-Mar no 3   
233 70-79 ge40 15-19 11-Sep yes 1   
234 60-69 ge40 30-34 0-2 no 3   
235 50-59 premeno 25-29 5-Mar yes 3   
236 40-49 premeno 25-29 0-2 no 2   
237 40-49 premeno 25-29 0-2 no 2   
238 30-39 premeno 35-39 0-2 no 3   
239 40-49 premeno 20-24 5-Mar yes 2   
240 60-69 ge40 20-24 5-Mar no 2   
241 40-49 premeno 15-19 15-17 yes 3   
242 50-59 ge40 25-29 8-Jun no 3   
243 50-59 ge40 20-24 5-Mar yes 3   
244 40-49 premeno 30-34 14-Dec yes 3   
245 30-39 premeno 30-34 11-Sep no 2   
246 30-39 premeno 15-19 8-Jun yes 3   
247 50-59 ge40 30-34 11-Sep yes 3   
248 60-69 ge40 35-39 8-Jun yes 3   
249 30-39 premeno 20-24 5-Mar yes 2   
250 40-49 premeno 25-29 0-2 no 3   
251 40-49 premeno 50-54 0-2 no 2   
252 30-39 premeno 40-44 0-2 no 1   
253 60-69 ge40 50-54 0-2 no 3   
254 40-49 premeno 30-34 0-2 yes 3   
255 40-49 premeno 30-34 8-Jun yes 3   
256 40-49 premeno 30-34 0-2 no 1   
257 40-49 premeno 20-24 5-Mar yes 2   
258 50-59 ge40 30-34 8-Jun yes 2   
259 50-59 ge40 30-34 5-Mar no 3   
260 60-69 ge40 25-29 5-Mar no 2   
261 40-49 ge40 25-29 14-Dec yes 3   
262 60-69 ge40 25-29 0-2 no 3   
263 50-59 lt40 20-24 0-2 no 1   
264 50-59 lt40 20-24 0-2 no 1   
265 30-39 premeno 35-39 11-Sep yes 3   
266 40-49 premeno 30-34 5-Mar yes 2   
267 60-69 ge40 20-24 24-26 yes 3   
268 30-39 premeno 35-39 0-2 no 3   
269 40-49 premeno 25-29 0-2 no 2   
270 50-59 ge40 30-34 8-Jun yes 3   
271 50-59 premeno 25-29 0-2 no 3   
272 40-49 premeno 15-19 0-2 yes 3   
273 60-69 ge40 30-34 0-2 yes 2   
274 60-69 ge40 30-34 5-Mar yes 2   
275 40-49 premeno 25-29 11-Sep yes 3   
276 30-39 premeno 25-29 8-Jun yes 3   
277 60-69 ge40 14-Oct 8-Jun yes 3   
278 50-59 premeno 35-39 15-17 yes 3   
279 50-59 ge40 40-44 8-Jun yes 3   
280 50-59 ge40 40-44 8-Jun yes 3   
281 30-39 premeno 30-34 0-2 no 2   
282 30-39 premeno 20-24 0-2 no 3   
283 60-69 ge40 20-24 0-2 no 1   
284 40-49 ge40 30-34 5-Mar no 3   
285 50-59 ge40 30-34 5-Mar no 3   
  
 Breast Breast-Quad Irradiat Class   
226 left left\_low no recurrence-events   
227 right left\_up yes recurrence-events   
228 right left\_low no recurrence-events   
229 right right\_up yes recurrence-events   
230 right left\_up no recurrence-events   
231 right left\_up no recurrence-events   
232 right left\_up yes recurrence-events   
233 left left\_low yes recurrence-events   
234 right left\_up yes recurrence-events   
235 left left\_low yes recurrence-events   
236 right left\_low no recurrence-events   
237 right left\_low no recurrence-events   
238 left left\_low no recurrence-events   
239 right right\_up yes recurrence-events   
240 left left\_low yes recurrence-events   
241 left left\_low no recurrence-events   
242 left left\_low yes recurrence-events   
243 right right\_up no recurrence-events   
244 left left\_up yes recurrence-events   
245 right left\_up yes recurrence-events   
246 left left\_low yes recurrence-events   
247 left right\_low yes recurrence-events   
248 left left\_low no recurrence-events   
249 left left\_low no recurrence-events   
250 left left\_up no recurrence-events   
251 right left\_low yes recurrence-events   
252 left left\_up no recurrence-events   
253 right left\_up no recurrence-events   
254 right right\_up no recurrence-events   
255 right left\_up no recurrence-events   
256 left left\_low yes recurrence-events   
257 left left\_low yes recurrence-events   
258 left right\_low yes recurrence-events   
259 right left\_up no recurrence-events   
260 right right\_up no recurrence-events   
261 left right\_low yes recurrence-events   
262 left left\_up no recurrence-events   
263 left left\_up no recurrence-events   
264 left left\_low no recurrence-events   
265 left left\_low no recurrence-events   
266 left right\_up no recurrence-events   
267 left left\_low yes recurrence-events   
268 left left\_low no recurrence-events   
269 left left\_low yes recurrence-events   
270 left right\_low no recurrence-events   
271 right left\_low yes recurrence-events   
272 right left\_up no recurrence-events   
273 right right\_up yes recurrence-events   
274 left central yes recurrence-events   
275 right left\_up no recurrence-events   
276 left right\_low yes recurrence-events   
277 left left\_up yes recurrence-events   
278 right right\_up no recurrence-events   
279 left left\_low yes recurrence-events   
280 left left\_low yes recurrence-events   
281 left left\_up no recurrence-events   
282 left left\_up yes recurrence-events   
283 right left\_up no recurrence-events   
284 left left\_low no recurrence-events   
285 left left\_low no recurrence-events

## **Data Redundancy**

## # This code was done by 223077593 - ORILIO NAOBEB and Nakaleke Fillemon 223098523

## duplicated\_rows = breast\_cancer[breast\_cancer.duplicated(keep=False)] # keep=False shows all duplicates

## print(duplicated\_rows)

## Age Range Menopause Tumor size Inv-nodes Node-caps Degree of malignancy \

## 3 60-69 ge40 15-19 0-2 no 2

## 5 60-69 ge40 15-19 0-2 no 2

## 6 50-59 premeno 25-29 0-2 no 2

## 7 60-69 ge40 20-24 0-2 no 1

## 9 40-49 premeno 20-24 0-2 no 2

## 21 60-69 ge40 15-19 0-2 no 2

## 37 50-59 ge40 15-19 0-2 no 1

## 65 40-49 premeno 14-Oct 0-2 no 1

## 67 50-59 ge40 20-24 0-2 no 3

## 78 50-59 premeno 25-29 0-2 no 2

## 81 60-69 ge40 14-Oct 0-2 no 1

## 91 50-59 ge40 20-24 0-2 no 3

## 97 60-69 ge40 15-19 0-2 no 2

## 111 50-59 ge40 15-19 0-2 no 1

## 112 40-49 premeno 20-24 0-2 no 2

## 113 40-49 premeno 14-Oct 0-2 no 1

## 116 30-39 premeno 15-19 0-2 no 1

## 120 60-69 ge40 20-24 0-2 no 1

## 142 60-69 ge40 14-Oct 0-2 no 1

## 171 30-39 premeno 15-19 0-2 no 1

## 204 50-59 premeno 25-29 0-2 no 2

## 207 50-59 premeno 25-29 0-2 no 2

## 236 40-49 premeno 25-29 0-2 no 2

## 237 40-49 premeno 25-29 0-2 no 2

## 238 30-39 premeno 35-39 0-2 no 3

## 268 30-39 premeno 35-39 0-2 no 3

## 279 50-59 ge40 40-44 8-Jun yes 3

## 280 50-59 ge40 40-44 8-Jun yes 3

## Breast Breast-Quad Irradiat Class

## 3 right left\_up no no-recurrence-events

## 5 left left\_low no no-recurrence-events

## 6 left left\_low no no-recurrence-events

## 7 left left\_low no no-recurrence-events

## 9 right left\_up no no-recurrence-events

## 21 left left\_low no no-recurrence-events

## 37 right central no no-recurrence-events

## 65 right left\_up no no-recurrence-events

## 67 left left\_up no no-recurrence-events

## 78 left left\_low no no-recurrence-events

## 81 left left\_up no no-recurrence-events

## 91 left left\_up no no-recurrence-events

## 97 right left\_up no no-recurrence-events

## 111 right central no no-recurrence-events

## 112 right left\_up no no-recurrence-events

## 113 right left\_up no no-recurrence-events

## 116 left left\_low no no-recurrence-events

## 120 left left\_low no no-recurrence-events

## 142 left left\_up no no-recurrence-events

## 171 left left\_low no no-recurrence-events

## 204 left right\_up no recurrence-events

## 207 left right\_up no recurrence-events

## 236 right left\_low no recurrence-events

## 237 right left\_low no recurrence-events

## 238 left left\_low no recurrence-events

## 268 left left\_low no recurrence-events

## 279 left left\_low yes recurrence-events

## 280 left left\_low yes recurrence-events

## # This code was done by 223077593 - ORILIO NAOBEB and Nakaleke Fillemon 223098523

## breast\_cancer\_cleaned = breast\_cancer.drop\_duplicates(keep='first') # Keeps the first occurrence

## # This code was done by 223077593 - ORILIO NAOBEB and Nakaleke Fillemon 223098523

## print(f"Number of duplicate rows after cleaning: {breast\_cancer\_cleaned.duplicated().sum()}")

## Number of duplicate rows after cleaning: 0

## **Typos**

# This code was done by 223077593 - ORILIO NAOBEB  
print(breast\_cancer\_cleaned['Inv-nodes'].unique())

['0-2' '8-Jun' '11-Sep' '5-Mar' '15-17' '14-Dec' '24-26']

# This code was done by 223077593 - ORILIO NAOBEB  
print(breast\_cancer\_cleaned['Tumor size'].unique())

['30-34' '20-24' '15-19' '0-4' '25-29' '50-54' '14-Oct' '40-44' '35-39'  
 '9-May' '45-49']

# This code was done by 223077593 - ORILIO NAOBEB  
# Define correct categories for Tumor size and Inv-nodes  
correct\_tumor\_sizes = ['0-4', '5-9', '10-14', '15-19', '20-24', '25-29', '30-34', '35-39', '40-44', '45-49', '50-54']  
correct\_inv\_nodes = ['0-2', '3-5', '6-8', '9-11', '12-14', '15-17', '18-20','21-23','24-26']  
  
# Function to apply fuzzy matching and replace incorrect values  
def correct\_values(column, correct\_values\_list):  
 return column.apply(lambda x: process.extractOne(x, correct\_values\_list)[0])  
  
# Use .loc to apply fuzzy matching on Tumor size column  
breast\_cancer\_cleaned.loc[:, 'Tumor size'] = correct\_values(breast\_cancer\_cleaned['Tumor size'], correct\_tumor\_sizes)  
  
# Use .loc to apply fuzzy matching on Inv-nodes column  
breast\_cancer\_cleaned.loc[:, 'Inv-nodes'] = correct\_values(breast\_cancer\_cleaned['Inv-nodes'], correct\_inv\_nodes)  
  
# Check corrected values  
print(breast\_cancer\_cleaned[['Tumor size', 'Inv-nodes']])

Tumor size Inv-nodes  
0 30-34 0-2  
1 20-24 0-2  
2 20-24 0-2  
3 15-19 0-2  
4 0-4 0-2  
.. ... ...  
281 30-34 0-2  
282 20-24 0-2  
283 20-24 0-2  
284 30-34 3-5  
285 30-34 3-5  
  
[272 rows x 2 columns]

# This code was done by 223077593 - ORILIO NAOBEB  
print(breast\_cancer\_cleaned['Inv-nodes'].unique())

['0-2' '6-8' '9-11' '3-5' '15-17' '12-14' '24-26']

# This code was done by 223077593 - ORILIO NAOBEB  
print(breast\_cancer\_cleaned['Tumor size'].unique())

['30-34' '20-24' '15-19' '0-4' '25-29' '50-54' '10-14' '40-44' '35-39'  
 '5-9' '45-49']

## **Outliers**

# This code was done by 223077593 - ORILIO NAOBEB  
#There is no need for outlier cleaning because the Degree of Malignancy only contains three distinct values (1, 2, and 3) with no outliers.   
#The remaining categorical data does not have outliers, as outlier detection is relevant only for numerical data.

# VISUALIZATION

# *# This code was done by 222129816 - TANAKA MUSHUKUTU*

# *# Display the first few rows and basic information about the dataset*

# *data\_info* ***=*** *breast\_cancer\_cleaned****.****info()*

# *data\_head* ***=*** *breast\_cancer\_cleaned****.****head()*

# *data\_info, data\_head*

# *<class 'pandas.core.frame.DataFrame'>*

# *Index: 272 entries, 0 to 285*

# *Data columns (total 10 columns):*

# *# Column Non-Null Count Dtype*

# *--- ------ -------------- -----*

# *0 Age Range 272 non-null object*

# *1 Menopause 272 non-null object*

# *2 Tumor size 272 non-null object*

# *3 Inv-nodes 272 non-null object*

# *4 Node-caps 272 non-null object*

# *5 Degree of malignancy 272 non-null int64*

# *6 Breast 272 non-null object*

# *7 Breast-Quad 272 non-null object*

# *8 Irradiat 272 non-null object*

# *9 Class 272 non-null object*

# *dtypes: int64(1), object(9)*

# *memory usage: 23.4+ KB*

# *Out[41]:*

# *(None,*

# *Age Range Menopause Tumor size Inv-nodes Node-caps Degree of malignancy \*

# *0 30-39 premeno 30-34 0-2 no 3*

# *1 40-49 premeno 20-24 0-2 no 2*

# *2 40-49 premeno 20-24 0-2 no 2*

# *3 60-69 ge40 15-19 0-2 no 2*

# *4 40-49 premeno 0-4 0-2 no 2*

# 

# *Breast Breast-Quad Irradiat Class*

# *0 left left\_low no no-recurrence-events*

# *1 right right\_up no no-recurrence-events*

# *2 left left\_low no no-recurrence-events*

# *3 right left\_up no no-recurrence-events*

# *4 right right\_low no no-recurrence-events )*

# *In [42]:*

# *# This code was done by 222129816 - TANAKA MUSHUKUTU*

# *# Pie Chart: Distribution of the target variable `Class`*

# *class\_distribution* ***=*** *breast\_cancer\_cleaned['Class']****.****value\_counts()*

# *plt****.****figure(figsize****=****(8, 8))*

# *plt****.****pie(class\_distribution, labels****=****class\_distribution****.****index, autopct****=****'%1.1f%%', startangle****=****140, colors****=****sns****.****color\_palette("pastel"))*

# *plt****.****title("Distribution of Breast Cancer Class (Recurrence vs. No Recurrence)")*

# *plt****.****show()*

# *# Bar Graph: Tumor size vs Degree of malignancy (using hue for different colors)*

# *plt****.****figure(figsize****=****(12, 6))*

# *sns****.****barplot(x****=****'Tumor size', y****=****'Degree of malignancy', data****=****breast\_cancer\_cleaned, hue****=****'Tumor size', palette****=****"viridis", errorbar****=None****)*

# *plt****.****title("Tumor Size vs Degree of Malignancy")*

# *plt****.****xticks(rotation****=****45)*

# *plt****.****legend([], [], frameon****=False****) # Hides the legend*

# *plt****.****show()*

# 

# *A graph of different colored bars Description automatically generated*

# *In [43]:*

# *# This code was done by 222129816 - TANAKA MUSHUKUTU*

# *# Interactive Visualization: Tumor size vs Age Range grouped by Menopause status*

# *fig* ***=*** *px****.****scatter(breast\_cancer\_cleaned, x****=****'Age Range', y****=****'Tumor size', color****=****'Menopause',*

# *hover\_data****=****['Degree of malignancy', 'Breast', 'Irradiat'],*

# *title****=****"Tumor Size vs Age Range Grouped by Menopause Status",*

# *labels****=****{'Age Range': 'Age Range', 'Tumor size': 'Tumor Size'})*

# *fig****.****show()*

# *# Normal Distribution: Visualizing `Degree of malignancy`*

# *plt****.****figure(figsize****=****(10, 6))*

# *sns****.****histplot(breast\_cancer\_cleaned['Degree of malignancy'], kde****=True****, color****=****'skyblue', bins****=****10)*

# *plt****.****title('Distribution of Degree of Malignancy')*

# *plt****.****xlabel('Degree of Malignancy')*

# *plt****.****ylabel('Frequency')*

# *plt****.****show()*

# *A graph with blue lines and numbers Description automatically generated*

# *In [44]:*

# *# This code was done by 222129816 - TANAKA MUSHUKUTU*

# *# Box plot: Comparing Degree of malignancy for different Breast Quadrants*

# *plt****.****figure(figsize****=****(12, 6))*

# *sns****.****boxplot(x****=****'Breast-Quad', y****=****'Degree of malignancy', data****=****breast\_cancer\_cleaned, hue****=****'Breast-Quad', palette****=****'coolwarm') # Using hue*

# *plt****.****title('Degree of Malignancy by Breast Quadrant')*

# *plt****.****xticks(rotation****=****45)*

# *plt****.****legend([], [], frameon****=False****) # Hide legend if not needed*

# *plt****.****show()*

# *A graph of different colored squares Description automatically generated*

# *In [45]:*

# *# This code was done by 223002291 - Beaven Muyoba*

# *# Count the occurrences of each unique value in the 'Age Range' column*

# *age\_range\_counts* ***=*** *breast\_cancer\_cleaned['Age Range']****.****value\_counts()*

# *# Plot a pie chart*

# *plt****.****figure(figsize****=****(8, 8))*

# *plt****.****pie(age\_range\_counts, labels****=****age\_range\_counts****.****index, autopct****=****'%1.1f%%', startangle****=****90)*

# *plt****.****title('Distribution of Age Range')*

# *plt****.****axis('equal') # Equal aspect ratio ensures that pie is drawn as a circle.*

# *# Show the pie chart*

# *plt****.****show()*

# 

# *In [857]:*

# *#223085065 Asser Naikuti*

# *# Mapping age ranges to midpoints*

# ***def*** *age\_range\_to\_midpoint(age\_range):*

# ***if*** *isinstance(age\_range, str): # Check if age\_range is a string*

# *start, end* ***=*** *map(int, age\_range****.****split('-'))*

# ***return*** *(start* ***+*** *end)* ***/*** *2*

# ***else****:*

# ***return*** *np****.****nan # Return NaN for non-string inputs*

# *# Convert 'Age Range' to midpoints using .loc (avoiding SettingWithCopyWarning)*

# *breast\_cancer\_cleaned* ***=*** *breast\_cancer\_cleaned****.****copy() # Create a deep copy to avoid the warning*

# *breast\_cancer\_cleaned****.****loc[:, 'Age Midpoint']* ***=*** *breast\_cancer\_cleaned['Age Range']****.****apply(age\_range\_to\_midpoint)*

# *# Now plot using midpoints*

# *plt****.****scatter(breast\_cancer\_cleaned['Age Midpoint'], breast\_cancer\_cleaned['Tumor size'])*

# *plt****.****xlabel('Age Midpoint')*

# *plt****.****ylabel('Tumor Size')*

# *plt****.****title('Scatter Plot of Tumor Size vs Age Midpoint')*

# *plt****.****show()*

# 

# *In [46]:*

# *# This code was done by 223002291 - Beaven Muyoba*

# *age\_range\_counts* ***=*** *breast\_cancer\_cleaned['Age Range']****.****value\_counts()*

# *# Plot a bar graph*

# *plt****.****figure(figsize****=****(10, 6))*

# *age\_range\_counts****.****plot(kind****=****'bar', color****=****'skyblue')*

# *# Adding titles and labels*

# *plt****.****title('Distribution of Age Range', fontsize****=****16)*

# *plt****.****xlabel('Age Range', fontsize****=****14)*

# *plt****.****ylabel('Count', fontsize****=****14)*

# *# Display the bar graph*

# *plt****.****xticks(rotation****=****45) # Rotate x-axis labels for better visibility*

# *plt****.****grid(axis****=****'y') # Optional: add grid lines for better readability*

# *plt****.****show()*

# *A graph of a distribution of age range Description automatically generated*

# *In [47]:*

# *# This code was done by 223033030 - JEROME JEG NANUSEB*

# *# Shows how the distribution of classes changes across different age ranges*

# *breast\_cancer\_cleaned\_grouped* ***=*** *breast\_cancer\_cleaned****.****groupby(['Age Range', 'Class'])****.****size()****.****unstack()*

# *breast\_cancer\_cleaned\_grouped****.****plot(kind****=****'bar', stacked****=True****, figsize****=****(12, 6))*

# *plt****.****title('Distribution of Classes Across Age Ranges')*

# *plt****.****xlabel('Age Range')*

# *plt****.****ylabel('Count')*

# *plt****.****legend(title****=****'Class', bbox\_to\_anchor****=****(1.05, 1), loc****=****'upper left')*

# *plt****.****tight\_layout()*

# *plt****.****show()*

# *A graph of blue and orange bars Description automatically generated*

# *In [48]:*

# *# This code was done by 223077593 - ORILIO NAOBEB*

# *breast\_cancer\_cleaned****.****hist(bins****=****15, color****=****'skyblue', edgecolor****=****'black', figsize****=****(10, 8))*

# *plt****.****suptitle('Histograms of Breast Cancer Data', fontsize****=****14) # Simple overall title*

# *plt****.****tight\_layout()*

# *plt****.****show()*

# *A graph with blue bars Description automatically generated*

# **Machine Learning**

## *# This code was done by Emma Muulyao 223066745 and Orilio Naobeb 223077593*

## *# Initialize LabelEncoder*

## *label\_encoder* ***=*** *LabelEncoder()*

## *# Identify all categorical columns in the dataset*

## *categorical\_columns* ***=*** *breast\_cancer\_cleaned****.****select\_dtypes(include****=****['object'])****.****columns*

## *# Apply LabelEncoder to all categorical columns*

## ***for*** *col* ***in*** *categorical\_columns:*

## *breast\_cancer\_cleaned****.****loc[:, col]* ***=*** *label\_encoder****.****fit\_transform(breast\_cancer\_cleaned[col])*

## *In [50]:*

## *# This code was done by Emma Muulyao 223066745 and Orilio Naobeb 223077593*

## *# Split the data into features (X) and target (Y)*

## *array* ***=*** *breast\_cancer\_cleaned****.****values*

## *X* ***=*** *array[:, 0:9]*

## *Y* ***=*** *array[:, 9]*

## *# Split data into training and validation sets*

## *validation\_size* ***=*** *0.4*

## *seed* ***=*** *7*

## *scoring* ***=*** *'accuracy'*

## *X\_train, X\_validation, Y\_train, Y\_validation* ***=*** *model\_selection****.****train\_test\_split(X, Y, test\_size****=****validation\_size, random\_state****=****seed)*

## *In [51]:*

## *# This code was done by Orilio Naobeb 223077593*

## *# Ensure that Y\_train has categorical values (Encode Y\_train)*

## *label\_encoder* ***=*** *LabelEncoder()*

## *Y\_train* ***=*** *label\_encoder****.****fit\_transform(Y\_train)*

## *Y\_validation* ***=*** *label\_encoder****.****transform(Y\_validation) # Ensure consistency for validation set*

## *In [52]:*

## *# This code was done by Emma Muulyao 223066745 and Orilio Naobeb 223077593*

## *models* ***=*** *[]*

## *models****.****append(('LR', LogisticRegression()))*

## *models****.****append(('LDA', LinearDiscriminantAnalysis()))*

## *models****.****append(('KNN', KNeighborsClassifier()))*

## *models****.****append(('CART', DecisionTreeClassifier()))*

## *models****.****append(('NB', GaussianNB()))*

## *models****.****append(('SVM', SVC()))*

## *models****.****append(('RF', RandomForestClassifier()))*

## *models****.****append(('GB', GradientBoostingClassifier()))*

## *# print(models)*

## *# evaluate each model in turn*

## *results* ***=*** *[]*

## *names* ***=*** *[]*

## ***for*** *name, model* ***in*** *models:*

## *kfold* ***=*** *model\_selection****.****KFold(n\_splits****=****13, random\_state****=****seed, shuffle****=True****)*

## *cv\_results* ***=*** *model\_selection****.****cross\_val\_score(model, X\_train, Y\_train, cv****=****kfold, scoring****=****scoring)*

## *results****.****append(cv\_results)*

## *names****.****append(name)*

## *msg* ***=*** *"%s: %f (%f)"* ***%*** *(name, cv\_results****.****mean(), cv\_results****.****std())*

## *print(msg)*

## *LR: 0.774162 (0.085047)*

## *LDA: 0.773669 (0.081565)*

## *KNN: 0.693294 (0.102990)*

## *CART: 0.657298 (0.096519)*

## *NB: 0.758383 (0.119856)*

## *SVM: 0.748028 (0.107158)*

## *RF: 0.729783 (0.122296)*

## *GB: 0.730769 (0.104215)*

## *In [53]:*

## *# This code was done by Loide Nambundunga 221060154 and Orilio Naobeb 223077593*

## *# Compare Algos*

## *fig* ***=*** *plt****.****figure()*

## *fig****.****suptitle('Algorthm Comparison')*

## *ax* ***=*** *fig****.****add\_subplot(111)*

## *plt****.****boxplot(results)*

## *ax****.****set\_xticklabels(names)*

## *plt****.****show()*

## 

## *In [54]:*

## *# This code was done by Orilio Naobeb 223077593*

## *# Logistic Regression*

## *lr* ***=*** *LogisticRegression(C****=****10.0, random\_state****=****41)*

## *lr****.****fit(X\_train, Y\_train)*

## *predictions\_lr* ***=*** *lr****.****predict(X\_validation)*

## *print("Logistic Regression Results:")*

## *print("Accuracy:", accuracy\_score(Y\_validation, predictions\_lr))*

## *print("Confusion Matrix:\n", confusion\_matrix(Y\_validation, predictions\_lr))*

## *print("Classification Report:\n", classification\_report(Y\_validation, predictions\_lr))*

## *Logistic Regression Results:*

## *Accuracy: 0.6972477064220184*

## *Confusion Matrix:*

## *[[68 3]*

## *[30 8]]*

## *Classification Report:*

## *precision recall f1-score support*

## *0 0.69 0.96 0.80 71*

## *1 0.73 0.21 0.33 38*

## *accuracy 0.70 109*

## *macro avg 0.71 0.58 0.57 109*

## *weighted avg 0.71 0.70 0.64 109*

## *In [55]:*

## *# This code was done by Orilio Naobeb 223077593*

## *rf* ***=*** *RandomForestClassifier(n\_estimators****=****150, max\_depth****=****7, min\_samples\_split****=****2, random\_state****=****41)*

## *rf****.****fit(X\_train, Y\_train)*

## *predictions* ***=*** *rf****.****predict(X\_validation)*

## *# Evaluate the RandomForest model*

## *print("Accuracy:", accuracy\_score(Y\_validation, predictions))*

## *print("\t=============================================\n")*

## *print("Confusion Matrix:\n", confusion\_matrix(Y\_validation, predictions))*

## *print("\t=============================================\n")*

## *print("Classification Report:\n", classification\_report(Y\_validation, predictions))*

## *Accuracy: 0.6697247706422018*

## *=============================================*

## *Confusion Matrix:*

## *[[65 6]*

## *[30 8]]*

## *=============================================*

## *Classification Report:*

## *precision recall f1-score support*

## *0 0.68 0.92 0.78 71*

## *1 0.57 0.21 0.31 38*

## *accuracy 0.67 109*

## *macro avg 0.63 0.56 0.55 109*

## *weighted avg 0.64 0.67 0.62 109*

## *In [56]:*

## *# This code was done by Orilio Naobeb 223077593*

## *# Linear Discriminant Analysis*

## *lda* ***=*** *LinearDiscriminantAnalysis()*

## *lda****.****fit(X\_train, Y\_train)*

## *predictions\_lda* ***=*** *lda****.****predict(X\_validation)*

## *print("\nLinear Discriminant Analysis Results:")*

## *print("Accuracy:", accuracy\_score(Y\_validation, predictions\_lda))*

## *print("Confusion Matrix:\n", confusion\_matrix(Y\_validation, predictions\_lda))*

## *print("Classification Report:\n", classification\_report(Y\_validation, predictions\_lda))*

## *Linear Discriminant Analysis Results:*

## *Accuracy: 0.6880733944954128*

## *Confusion Matrix:*

## *[[67 4]*

## *[30 8]]*

## *Classification Report:*

## *precision recall f1-score support*

## *0 0.69 0.94 0.80 71*

## *1 0.67 0.21 0.32 38*

## *accuracy 0.69 109*

## *macro avg 0.68 0.58 0.56 109*

## *weighted avg 0.68 0.69 0.63 109*

## *In [57]:*

## *# This code was done by Orilio Naobeb 223077593*

## *# K-Nearest Neighbors*

## *knn* ***=*** *KNeighborsClassifier(n\_neighbors****=****3, weights****=****'uniform')*

## *knn****.****fit(X\_train, Y\_train)*

## *predictions\_knn* ***=*** *knn****.****predict(X\_validation)*

## *print("\nK-Nearest Neighbors Results:")*

## *print("Accuracy:", accuracy\_score(Y\_validation, predictions\_knn))*

## *print("Confusion Matrix:\n", confusion\_matrix(Y\_validation, predictions\_knn))*

## *print("Classification Report:\n", classification\_report(Y\_validation, predictions\_knn))*

## *K-Nearest Neighbors Results:*

## *Accuracy: 0.6605504587155964*

## *Confusion Matrix:*

## *[[66 5]*

## *[32 6]]*

## *Classification Report:*

## *precision recall f1-score support*

## *0 0.67 0.93 0.78 71*

## *1 0.55 0.16 0.24 38*

## *accuracy 0.66 109*

## *macro avg 0.61 0.54 0.51 109*

## *weighted avg 0.63 0.66 0.59 109*

## *In [58]:*

## *# This code was done by Orilio Naobeb 223077593*

## *# Decision Tree Classifier (CART)*

## *cart* ***=*** *DecisionTreeClassifier(max\_depth****=****7, min\_samples\_split****=****4, random\_state****=****41)*

## *cart****.****fit(X\_train, Y\_train)*

## *predictions\_cart* ***=*** *cart****.****predict(X\_validation)*

## *print("\nDecision Tree (CART) Results:")*

## *print("Accuracy:", accuracy\_score(Y\_validation, predictions\_cart))*

## *print("Confusion Matrix:\n", confusion\_matrix(Y\_validation, predictions\_cart))*

## *print("Classification Report:\n", classification\_report(Y\_validation, predictions\_cart))*

## *Decision Tree (CART) Results:*

## *Accuracy: 0.6422018348623854*

## *Confusion Matrix:*

## *[[60 11]*

## *[28 10]]*

## *Classification Report:*

## *precision recall f1-score support*

## *0 0.68 0.85 0.75 71*

## *1 0.48 0.26 0.34 38*

## *accuracy 0.64 109*

## *macro avg 0.58 0.55 0.55 109*

## *weighted avg 0.61 0.64 0.61 109*

## *In [59]:*

## *# This code was done by Orilio Naobeb 223077593*

## *# Gaussian Naive Bayes*

## *nb* ***=*** *GaussianNB()*

## *nb****.****fit(X\_train, Y\_train)*

## *predictions\_nb* ***=*** *nb****.****predict(X\_validation)*

## *print("\nGaussian Naive Bayes Results:")*

## *print("Accuracy:", accuracy\_score(Y\_validation, predictions\_nb))*

## *print("Confusion Matrix:\n", confusion\_matrix(Y\_validation, predictions\_nb))*

## *print("Classification Report:\n", classification\_report(Y\_validation, predictions\_nb))*

## *Gaussian Naive Bayes Results:*

## *Accuracy: 0.7247706422018348*

## *Confusion Matrix:*

## *[[63 8]*

## *[22 16]]*

## *Classification Report:*

## *precision recall f1-score support*

## *0 0.74 0.89 0.81 71*

## *1 0.67 0.42 0.52 38*

## *accuracy 0.72 109*

## *macro avg 0.70 0.65 0.66 109*

## *weighted avg 0.72 0.72 0.71 109*

## *In [60]:*

## *# This code was done by Orilio Naobeb 223077593*

## *# Support Vector Machine (SVM)*

## *svm* ***=*** *SVC(C****=****1.0, kernel****=****'linear', random\_state****=****41)*

## *svm****.****fit(X\_train, Y\_train)*

## *predictions\_svm* ***=*** *svm****.****predict(X\_validation)*

## *print("\nSupport Vector Machine (SVM) Results:")*

## *print("Accuracy:", accuracy\_score(Y\_validation, predictions\_svm))*

## *print("Confusion Matrix:\n", confusion\_matrix(Y\_validation, predictions\_svm))*

## *print("Classification Report:\n", classification\_report(Y\_validation, predictions\_svm))*

## *Support Vector Machine (SVM) Results:*

## *Accuracy: 0.6880733944954128*

## *Confusion Matrix:*

## *[[69 2]*

## *[32 6]]*

## *Classification Report:*

## *precision recall f1-score support*

## *0 0.68 0.97 0.80 71*

## *1 0.75 0.16 0.26 38*

## *accuracy 0.69 109*

## *macro avg 0.72 0.56 0.53 109*

## *weighted avg 0.71 0.69 0.61 109*

## *In [61]:*

## *# This code was done by Orilio Naobeb 223077593*

## *# Gradient Boosting (GB)*

## *gb* ***=*** *GradientBoostingClassifier(n\_estimators****=****150, learning\_rate****=****0.01, random\_state****=****41)*

## *gb****.****fit(X\_train, Y\_train)*

## *predictions\_gb* ***=*** *gb****.****predict(X\_validation)*

## *print("\nGradient Boosting Results:")*

## *print("Accuracy:", accuracy\_score(Y\_validation, predictions\_gb))*

## *print("Confusion Matrix:\n", confusion\_matrix(Y\_validation, predictions\_gb))*

## *print("Classification Report:\n", classification\_report(Y\_validation, predictions\_gb))*

## *Gradient Boosting Results:*

## *Accuracy: 0.6697247706422018*

## *Confusion Matrix:*

## *[[67 4]*

## *[32 6]]*

## *Classification Report:*

## *precision recall f1-score support*

## *0 0.68 0.94 0.79 71*

## *1 0.60 0.16 0.25 38*

## *accuracy 0.67 109*

## *macro avg 0.64 0.55 0.52 109*

## *weighted avg 0.65 0.67 0.60 109*

## *In [62]:*

## *# This code was done by Orilio Naobeb 223077593*

## *print("Shape of X:", X****.****shape) # Check the number of samples and features*

## *print("Shape of y:", Y****.****shape) # Check the number of samples*

## *Shape of X: (272, 9)*

## *Shape of y: (272,)*

## *In [63]:*

## *# This code was done by Orilio Naobeb 223077593*

## *# Initialize LabelEncoder*

## *label\_encoder* ***=*** *LabelEncoder()*

## *# Encode the target variable (Class)*

## *y* ***=*** *label\_encoder****.****fit\_transform(breast\_cancer\_cleaned['Class'])*

## *# Define your features*

## *X* ***=*** *breast\_cancer\_cleaned****.****drop(columns****=****['Class']) # Replace with actual feature columns if needed*

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

## *X\_train, X\_test, y\_train, y\_test* ***=*** *train\_test\_split(X, y, test\_size****=****0.2, random\_state****=****42)*

## *# Train the models*

## *rf* ***=*** *RandomForestClassifier()*

## *rf****.****fit(X\_train, y\_train)*

## *ld* ***=*** *LinearDiscriminantAnalysis()*

## *ld****.****fit(X\_train, y\_train)*

## *knn* ***=*** *KNeighborsClassifier()*

## *knn****.****fit(X\_train, y\_train)*

## *dt* ***=*** *DecisionTreeClassifier()*

## *dt****.****fit(X\_train, y\_train)*

## *lr* ***=*** *LogisticRegression()*

## *lr****.****fit(X\_train, y\_train)*

## *nb* ***=*** *GaussianNB()*

## *nb****.****fit(X\_train, y\_train)*

## *svm* ***=*** *SVC()*

## *svm****.****fit(X\_train, y\_train)*

## *gb* ***=*** *GradientBoostingClassifier()*

## *gb****.****fit(X\_train, y\_train)*

## *Out[63]:*

## *GradientBoostingClassifier()*

## ***In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook. On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.***

## *In [64]:*

## *# This code was done by Orilio Naobeb 223077593*

## *print(metrics****.****accuracy\_score(y, rf****.****predict(X)))*

## *print(metrics****.****accuracy\_score(y, ld****.****predict(X)))*

## *print(metrics****.****accuracy\_score(y, knn****.****predict(X)))*

## *print(metrics****.****accuracy\_score(y, dt****.****predict(X)))*

## *print(metrics****.****accuracy\_score(y, lr****.****predict(X)))*

## *print(metrics****.****accuracy\_score(y, nb****.****predict(X)))*

## *print(metrics****.****accuracy\_score(y, svm****.****predict(X)))*

## *print(metrics****.****accuracy\_score(y, gb****.****predict(X)))*

## *0.9301470588235294*

## *0.7389705882352942*

## *0.7720588235294118*

## *0.9080882352941176*

## *0.75*

## *0.7352941176470589*

## *0.7573529411764706*

## *0.8676470588235294*

## *In [65]:*

## *# This code was done by Orilio Naobeb 223077593*

## *print(X****.****shape)*

## *print(y****.****shape)*

## *(272, 9)*

## *(272,)*

## *In [66]:*

## *# This code was done by Orilio Naobeb 223077593*

## *X\_train, X\_test, y\_train, y\_test* ***=*** *train\_test\_split(X, y, test\_size****=****0.2, random\_state****=****42) # Adjust test\_size as needed*

## *# Random state is given any random integer value so that each time we train and test our model the values remain same.*

## *In [67]:*

## *# This code was done by Orilio Naobeb 223077593*

## *print(X\_train****.****shape)*

## *print(X\_test****.****shape)*

## *(217, 9)*

## *(55, 9)*

## *In [68]:*

## *# This code was done by Orilio Naobeb 223077593*

## *print(y\_train****.****shape)*

## *print(y\_test****.****shape)*

## *(217,)*

## *(55,)*

## *In [69]:*

## *# This code was done by Orilio Naobeb 223077593*

## *knn* ***=*** *KNeighborsClassifier(n\_neighbors****=****5)*

## *knn****.****fit(X\_train, y\_train)*

## *Out[69]:*

## *KNeighborsClassifier()*

## ***In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook. On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.***

## *In [70]:*

## *# This code was done by Orilio Naobeb 223077593*

## *y\_predict* ***=*** *knn****.****predict(X\_test)*

## *print(y\_predict)*

## *[0 0 0 0 1 0 0 0 0 1 0 0 1 0 0 0 0 1 1 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0*

## *0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]*

## *In [71]:*

## *# This code was done by Orilio Naobeb 223077593*

## *print(metrics****.****accuracy\_score(y\_test, y\_predict))*

## *print(metrics****.****confusion\_matrix(y\_test, y\_predict))*

## *0.6909090909090909*

## *[[34 4]*

## *[13 4]]*

## *In [72]:*

## *# This code was done by Orilio Naobeb 223077593*

## *rf* ***=*** *RandomForestClassifier(n\_estimators****=****150, max\_depth****=****7, min\_samples\_split****=****2, random\_state****=****41)*

## *rf****.****fit(X\_train, y\_train)*

## *Out[72]:*

## *RandomForestClassifier(max\_depth=7, n\_estimators=150, random\_state=41)*

## ***In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook. On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.***

## *In [73]:*

## *# This code was done by Orilio Naobeb 223077593*

## *y\_predict* ***=*** *rf****.****predict(X\_test)*

## *print(y\_predict)*

## *[0 0 0 0 0 1 1 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 0*

## *0 0 1 0 0 0 0 0 0 0 0 0 1 0 1 1 0 0]*

## *In [74]:*

## *# This code was done by Orilio Naobeb 223077593*

## *print(metrics****.****accuracy\_score(y\_test, y\_predict))*

## *print(metrics****.****confusion\_matrix(y\_test, y\_predict))*

## *0.6363636363636364*

## *[[32 6]*

## *[14 3]]*

## *In [75]:*

## *# This code was done by Orilio Naobeb 223077593*

## *ld* ***=*** *LinearDiscriminantAnalysis()*

## *ld****.****fit(X\_train, y\_train)*

## *Out[75]:*

## *LinearDiscriminantAnalysis()*

## ***In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook. On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.***

## *In [76]:*

## *# This code was done by Orilio Naobeb 223077593*

## *y\_predict* ***=*** *ld****.****predict(X\_test)*

## *print(y\_predict)*

## *[0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0*

## *0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0]*

## *In [77]:*

## *# This code was done by Orilio Naobeb 223077593*

## *print(metrics****.****accuracy\_score(y\_test, y\_predict))*

## *print(metrics****.****confusion\_matrix(y\_test, y\_predict))*

## *0.7272727272727273*

## *[[36 2]*

## *[13 4]]*

## *In [78]:*

## *# This code was done by Orilio Naobeb 223077593*

## *dt* ***=*** *DecisionTreeClassifier(max\_depth****=****7, min\_samples\_split****=****4, random\_state****=****41)*

## *dt****.****fit(X\_train, y\_train)*

## *Out[78]:*

## *DecisionTreeClassifier(max\_depth=7, min\_samples\_split=4, random\_state=41)*

## ***In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook. On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.***

## *In [79]:*

## *# This code was done by Orilio Naobeb 223077593*

## *y\_predict* ***=*** *dt****.****predict(X\_test)*

## *print(y\_predict)*

## *[0 0 0 0 1 0 0 0 0 0 0 0 0 0 1 0 0 0 1 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0*

## *0 1 1 0 1 0 0 0 0 0 0 0 1 0 1 1 0 0]*

## *In [80]:*

## *# This code was done by Orilio Naobeb 223077593*

## *print(metrics****.****accuracy\_score(y\_test, y\_predict))*

## *print(metrics****.****confusion\_matrix(y\_test, y\_predict))*

## *0.5818181818181818*

## *[[30 8]*

## *[15 2]]*

## *In [81]:*

## *# This code was done by Orilio Naobeb 223077593 and Asser Naikuti 223085065*

## *lr* ***=*** *LogisticRegression()*

## *lr****.****fit(X\_train, y\_train)*

## *Out[81]:*

## *LogisticRegression()*

## ***In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook. On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.***

## *In [82]:*

## *# This code was done by Orilio Naobeb 223077593 and Asser Naikuti 223085065*

## *y\_predict* ***=*** *lr****.****predict(X\_test)*

## *print(y\_predict)*

## *[0 0 0 0 0 0 0 0 0 1 0 0 1 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0*

## *0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0]*

## *In [83]:*

## *# This code was done by Orilio Naobeb 223077593 and Asser Naikuti 223085065*

## *print(metrics****.****accuracy\_score(y\_test, y\_predict))*

## *print(metrics****.****confusion\_matrix(y\_test, y\_predict))*

## *0.7454545454545455*

## *[[36 2]*

## *[12 5]]*

## *In [84]:*

## *# This code was done by Orilio Naobeb 223077593*

## *nb* ***=*** *GaussianNB()*

## *nb****.****fit(X\_train, y\_train)*

## *Out[84]:*

## *GaussianNB()*

## ***In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook. On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.***

## *In [85]:*

## *# This code was done by Orilio Naobeb 223077593*

## *y\_predict* ***=*** *nb****.****predict(X\_test)*

## *print(y\_predict)*

## *[0 0 0 0 0 0 0 0 1 1 0 0 1 0 0 0 0 1 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0*

## *0 1 1 0 0 0 0 0 0 0 0 0 0 0 1 1 0 0]*

## *In [87]:*

## *# This code was done by Orilio Naobeb 223077593*

## *print(metrics****.****accuracy\_score(y\_test, y\_predict))*

## *print(metrics****.****confusion\_matrix(y\_test, y\_predict))*

## *0.6909090909090909*

## *[[33 5]*

## *[12 5]]*

## *In [88]:*

## *# This code was done by Orilio Naobeb 223077593*

## *svm* ***=*** *SVC()*

## *svm****.****fit(X\_train, y\_train)*

## *Out[88]:*

## *SVC()*

## ***In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook. On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.***

## *In [89]:*

## *# This code was done by Orilio Naobeb 223077593*

## *y\_predict* ***=*** *svm****.****predict(X\_test)*

## *print(y\_predict)*

## *[0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0*

## *0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]*

## *In [90]:*

## *# This code was done by Orilio Naobeb 223077593*

## *print(metrics****.****accuracy\_score(y\_test, y\_predict))*

## *print(metrics****.****confusion\_matrix(y\_test, y\_predict))*

## *0.7090909090909091*

## *[[36 2]*

## *[14 3]]*

## *In [91]:*

## *# This code was done by Orilio Naobeb 223077593*

## *gb* ***=*** *GradientBoostingClassifier(n\_estimators****=****150, learning\_rate****=****0.01, random\_state****=****41)*

## *gb****.****fit(X\_train, y\_train)*

## *Out[91]:*

## *GradientBoostingClassifier(learning\_rate=0.01, n\_estimators=150,*

## *random\_state=41)*

## ***In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook. On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.***

## *In [92]:*

## *# This code was done by Orilio Naobeb 223077593*

## *y\_predict* ***=*** *gb****.****predict(X\_test)*

## *print(y\_predict)*

## *[0 0 0 0 1 0 1 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0*

## *0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 1 0 0]*

## *In [93]:*

## *# This code was done by Orilio Naobeb 223077593*

## *print(metrics****.****accuracy\_score(y\_test, y\_predict))*

## *print(metrics****.****confusion\_matrix(y\_test, y\_predict))*

## *0.7454545454545455*

## *[[36 2]*

## *[12 5]]*

## ***Machine Learning Conclusion***

## ***We selected Logistic Regression for the breast cancer dataset due to its superior performance during both cross-validation and actual training. Logistic Regression is well-suited for binary classification problems, offering robustness in handling linear relationships. LDA, on the other hand, is particularly effective when the class distributions are normally distributed and separable. Given these factors, Logistic Regression emerged as the most reliable model for accurately classifying the dataset.***

## 6. Conclusion:

In conclusion, this breast cancer recurrence prediction project has demonstrated the significant potential of data analytics and machine learning in enhancing cancer prognosis and patient care. Through rigorous analysis of patient characteristics and medical factors, our team has developed predictive models that show promise in identifying individuals at higher risk of breast cancer recurrence.

The project's findings, including the identification of key factors influencing recurrence and the development of machine learning models with accuracies ranging from 73% to 77%, underscore the value of data-driven approaches in healthcare. These insights have the potential to assist medical professionals in tailoring follow-up care, informing treatment decisions, and ultimately improving patient outcomes.

The collaborative nature of this project highlighted the importance of interdisciplinary teamwork in tackling complex healthcare challenges. By combining diverse skills and perspectives, we were able to navigate the intricacies of data analysis and machine learning, while maintaining a focus on the project's potential real-world impact.

As we reflect on this project, we are reminded that behind every data point lies a human story. This understanding reinforces the significance of our work and its potential to make a meaningful difference in people's lives.

In essence, this project not only contributes to the field of breast cancer research but also serves as a testament to the power of data science in addressing critical healthcare challenges. As we conclude this project, we remain committed to furthering this important work, recognizing that our efforts may one day play a role in improving the lives of those affected by breast cancer.

7. Github Repository Link:

https://github.com/DrGitman/DTA621-Project.git