

BREAST CANCER DETECTION BY MUHAMMAD AHSAN



```
[296]: #Importing Important Libraries

import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline

#Algorithms
#Random Forest
from sklearn.ensemble import RandomForestClassifier
#K Neighbor
from sklearn.neighbors import KNeighborsClassifier
#SVM
from sklearn.svm import SVC
# Naive Bayes
from sklearn.naive_bayes import GaussianNB
#Decision Tree
from sklearn.tree import DecisionTreeClassifier
#Logistic Regression
from sklearn.linear_model import LogisticRegression

import warnings
warnings.filterwarnings("ignore")
```

Loading and displaying data

```
[297]: MyData= pd.read_csv("breast-cancer.csv")
MyData.head(4)
```

```
[297]:
```

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	\
0	842302	M	17.99	10.38	122.80	1001.0	
1	842517	M	20.57	17.77	132.90	1326.0	
2	84300903	M	19.69	21.25	130.00	1203.0	
3	84348301	M	11.42	20.38	77.58	386.1	

	smoothness_mean	compactness_mean	concavity_mean	concave points_mean	\
0	0.11840	0.27760	0.3001	0.14710	
1	0.08474	0.07864	0.0869	0.07017	
2	0.10960	0.15990	0.1974	0.12790	
3	0.14250	0.28390	0.2414	0.10520	

	... radius_worst	texture_worst	perimeter_worst	area_worst	\
0	... 25.38	17.33	184.60	2019.0	
1	... 24.99	23.41	158.80	1956.0	
2	... 23.57	25.53	152.50	1709.0	
3	... 14.91	26.50	98.87	567.7	

	smoothness_worst	compactness_worst	concavity_worst	concave points_worst	\
0	0.1622	0.6656	0.7119	0.2654	
1	0.1238	0.1866	0.2416	0.1860	
2	0.1444	0.4245	0.4504	0.2430	
3	0.2098	0.8663	0.6869	0.2575	

	symmetry_worst	fractal_dimension_worst
0	0.4601	0.11890
1	0.2750	0.08902
2	0.3613	0.08758
3	0.6638	0.17300

[4 rows x 32 columns]

0.1 INTRODUCTION

Breast cancer is a global health issue that impacts numerous individuals and their loved ones, emphasizing the need for early detection and precise diagnosis to enhance treatment effectiveness and patient outcomes. Over the past few years, the application of machine learning methods has demonstrated considerable potential in supporting healthcare practitioners with breast cancer diagnosis, offering a promising avenue for improved medical decision-making.

This project aims to analyze different features extracted from breast cancer patients in order to develop an accurate and efficient breast cancer prediction model. We will be using Supervised learning models which utilizes labeled data to learn patterns and make predictions. By training these models on a dataset of known breast cancer cases, we can develop algorithms that can accurately classify new, unseen instances as either Malignant (cancerous) or Benign (non-cancerous)

The primary aim of this project is to assess and compare the effectiveness of various supervised learning algorithms, including logistic regression, support vector machines, decision trees, random forests, SVM, and Naive Bayes. The evaluation will be based on metrics such as accuracy, precision, and recall. The results obtained from this study carry substantial importance in the realm of breast cancer diagnosis, potentially impacting the advancement of diagnostic methods in a meaningful way.

0.1.1 DATASET EXPLAINED

The kaggle dataset for breast cancer analysis consists of 569 rows and 32 columns. It consists of various features that are relevant to diagnosing breast cancer. Each data entry is identified by a unique “id” and is accompanied by a “diagnosis” label (target variable), indicating whether the breast mass is classified as Malignant (cancerous) or Benign (non-cancerous). The dataset encompasses informative features such as “radius_mean,” which represents the average distance from the center to points on the breast mass’s perimeter, providing insights into its size. “Texture_mean” signifies the variations in gray levels within the mass, serving as an indicator of textural characteristics. “Perimeter_mean” denotes the boundary length around the tumor, reflecting its extent. The “area_mean” feature measures the size of the mass in terms of the number of pixels, potentially indicating the presence of malignancy for larger areas.

“Smoothness_mean” characterizes the local variation in radius lengths, providing an average measure of smoothness within the mass. “Compactness_mean” quantifies how closely the points on the mass are packed together, conveying its compactness. “Concavity_mean” reflects the severity of concave regions within the mass, while “concave points_mean” signifies the number of such concave portions. The dataset also includes corresponding features for the “worst” and “standard error” values of the mean features mentioned above. For example, “radius_worst” represents the largest observed radius value within the breast mass, while “radius_se” represents the standard error associated with the radius measurement.

In addition, the dataset incorporates features related to smoothness, compactness, concavity, and concave points in their worst and standard error variations, further enriching the available information. Other relevant features like symmetry and fractal dimension are also present in the dataset, providing insights into the symmetry and complexity of the breast mass.

By analyzing these features and their variations, valuable insights can be gained regarding the characteristics of breast masses and their potential correlation with malignant or benign classifications.

0.1.2 PRELIMINARY ANALYSIS

0.1.3 Pre Processing

```
[298]: MyData.shape
```

```
[298]: (569, 32)
```

```
[299]: print(MyData.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 569 entries, 0 to 568
Data columns (total 32 columns):
#   Column                Non-Null Count  Dtype
---  -
0   id                    569 non-null   int64
1   diagnosis              569 non-null   object
2   radius_mean           569 non-null   float64
3   texture_mean          569 non-null   float64
4   perimeter_mean        569 non-null   float64
```

5	area_mean	569	non-null	float64
6	smoothness_mean	569	non-null	float64
7	compactness_mean	569	non-null	float64
8	concavity_mean	569	non-null	float64
9	concave points_mean	569	non-null	float64
10	symmetry_mean	569	non-null	float64
11	fractal_dimension_mean	569	non-null	float64
12	radius_se	569	non-null	float64
13	texture_se	569	non-null	float64
14	perimeter_se	569	non-null	float64
15	area_se	569	non-null	float64
16	smoothness_se	569	non-null	float64
17	compactness_se	569	non-null	float64
18	concavity_se	569	non-null	float64
19	concave points_se	569	non-null	float64
20	symmetry_se	569	non-null	float64
21	fractal_dimension_se	569	non-null	float64
22	radius_worst	569	non-null	float64
23	texture_worst	569	non-null	float64
24	perimeter_worst	569	non-null	float64
25	area_worst	569	non-null	float64
26	smoothness_worst	569	non-null	float64
27	compactness_worst	569	non-null	float64
28	concavity_worst	569	non-null	float64
29	concave points_worst	569	non-null	float64
30	symmetry_worst	569	non-null	float64
31	fractal_dimension_worst	569	non-null	float64

dtypes: float64(30), int64(1), object(1)

memory usage: 142.4+ KB

None

The datatypes are fine except Diagnosis target variable which we will be changing to binary 1 and 0 later

```
[300]: #checking Null Values
MyData.isna().sum()
```

```
[300]: id                0
diagnosis              0
radius_mean           0
texture_mean          0
perimeter_mean        0
area_mean             0
smoothness_mean       0
compactness_mean      0
concavity_mean        0
concave points_mean   0
symmetry_mean         0
```

```

fractal_dimension_mean    0
radius_se                 0
texture_se                0
perimeter_se              0
area_se                   0
smoothness_se             0
compactness_se            0
concavity_se              0
concave points_se         0
symmetry_se               0
fractal_dimension_se      0
radius_worst              0
texture_worst              0
perimeter_worst           0
area_worst                0
smoothness_worst          0
compactness_worst         0
concavity_worst           0
concave points_worst      0
symmetry_worst            0
fractal_dimension_worst   0
dtype: int64

```

There is no null value in the dataset

```

[301]: #Unique rows in each feature
MyData.nunique()

```

```

[301]: id                569
       diagnosis          2
       radius_mean       456
       texture_mean      479
       perimeter_mean    522
       area_mean         539
       smoothness_mean   474
       compactness_mean  537
       concavity_mean    537
       concave points_mean 542
       symmetry_mean     432
       fractal_dimension_mean 499
       radius_se         540
       texture_se        519
       perimeter_se      533
       area_se           528
       smoothness_se     547
       compactness_se    541
       concavity_se      533

```

```

concave points_se      507
symmetry_se            498
fractal_dimension_se   545
radius_worst           457
texture_worst          511
perimeter_worst        514
area_worst             544
smoothness_worst       411
compactness_worst      529
concavity_worst        539
concave points_worst   492
symmetry_worst         500
fractal_dimension_worst 535
dtype: int64

```

Diagnosis column has 2 unique values, indicating the two categories used to classify breast cancer instances, Malignant and Benign. Other columns has varying numbers of unique values, reflecting the diversity and range of measurements or features extracted from breast cancer data.

```

[302]: #checking for duplicates
MyData.duplicated().sum()

```

```

[302]: 0

```

There are no duplicates in the dataset

```

[303]: #dropping Id column as we don't need it in our analysis
MyData = MyData.drop("id", axis = 1)
MyData.head()

```

```

[303]:  diagnosis  radius_mean  texture_mean  perimeter_mean  area_mean  \
0           M         17.99         10.38         122.80       1001.0
1           M         20.57         17.77         132.90       1326.0
2           M         19.69         21.25         130.00       1203.0
3           M         11.42         20.38          77.58        386.1
4           M         20.29         14.34         135.10       1297.0

      smoothness_mean  compactness_mean  concavity_mean  concave points_mean  \
0           0.11840         0.27760         0.3001         0.14710
1           0.08474         0.07864         0.0869         0.07017
2           0.10960         0.15990         0.1974         0.12790
3           0.14250         0.28390         0.2414         0.10520
4           0.10030         0.13280         0.1980         0.10430

      symmetry_mean  ...  radius_worst  texture_worst  perimeter_worst  \
0           0.2419  ...         25.38         17.33         184.60
1           0.1812  ...         24.99         23.41         158.80
2           0.2069  ...         23.57         25.53         152.50

```

3	0.2597	...	14.91	26.50	98.87
4	0.1809	...	22.54	16.67	152.20

	area_worst	smoothness_worst	compactness_worst	concavity_worst	\
0	2019.0	0.1622	0.6656	0.7119	
1	1956.0	0.1238	0.1866	0.2416	
2	1709.0	0.1444	0.4245	0.4504	
3	567.7	0.2098	0.8663	0.6869	
4	1575.0	0.1374	0.2050	0.4000	

	concave points_worst	symmetry_worst	fractal_dimension_worst
0	0.2654	0.4601	0.11890
1	0.1860	0.2750	0.08902
2	0.2430	0.3613	0.08758
3	0.2575	0.6638	0.17300
4	0.1625	0.2364	0.07678

[5 rows x 31 columns]

0.1.4 Data Exploratory Analysis

```
[304]: # Descriptive Analysis
MyData
MyData.describe()
```

```
[304]:
```

	radius_mean	texture_mean	perimeter_mean	area_mean	\
count	569.000000	569.000000	569.000000	569.000000	
mean	14.127292	19.289649	91.969033	654.889104	
std	3.524049	4.301036	24.298981	351.914129	
min	6.981000	9.710000	43.790000	143.500000	
25%	11.700000	16.170000	75.170000	420.300000	
50%	13.370000	18.840000	86.240000	551.100000	
75%	15.780000	21.800000	104.100000	782.700000	
max	28.110000	39.280000	188.500000	2501.000000	

	smoothness_mean	compactness_mean	concavity_mean	concave points_mean	\
count	569.000000	569.000000	569.000000	569.000000	
mean	0.096360	0.104341	0.088799	0.048919	
std	0.014064	0.052813	0.079720	0.038803	
min	0.052630	0.019380	0.000000	0.000000	
25%	0.086370	0.064920	0.029560	0.020310	
50%	0.095870	0.092630	0.061540	0.033500	
75%	0.105300	0.130400	0.130700	0.074000	
max	0.163400	0.345400	0.426800	0.201200	

	symmetry_mean	fractal_dimension_mean	...	radius_worst	\
--	---------------	------------------------	-----	--------------	---

count	569.000000	569.000000	...	569.000000
mean	0.181162	0.062798	...	16.269190
std	0.027414	0.007060	...	4.833242
min	0.106000	0.049960	...	7.930000
25%	0.161900	0.057700	...	13.010000
50%	0.179200	0.061540	...	14.970000
75%	0.195700	0.066120	...	18.790000
max	0.304000	0.097440	...	36.040000

	texture_worst	perimeter_worst	area_worst	smoothness_worst	\
count	569.000000	569.000000	569.000000	569.000000	
mean	25.677223	107.261213	880.583128	0.132369	
std	6.146258	33.602542	569.356993	0.022832	
min	12.020000	50.410000	185.200000	0.071170	
25%	21.080000	84.110000	515.300000	0.116600	
50%	25.410000	97.660000	686.500000	0.131300	
75%	29.720000	125.400000	1084.000000	0.146000	
max	49.540000	251.200000	4254.000000	0.222600	

	compactness_worst	concavity_worst	concave	points_worst	\
count	569.000000	569.000000		569.000000	
mean	0.254265	0.272188		0.114606	
std	0.157336	0.208624		0.065732	
min	0.027290	0.000000		0.000000	
25%	0.147200	0.114500		0.064930	
50%	0.211900	0.226700		0.099930	
75%	0.339100	0.382900		0.161400	
max	1.058000	1.252000		0.291000	

	symmetry_worst	fractal_dimension_worst
count	569.000000	569.000000
mean	0.290076	0.083946
std	0.061867	0.018061
min	0.156500	0.055040
25%	0.250400	0.071460
50%	0.282200	0.080040
75%	0.317900	0.092080
max	0.663800	0.207500

[8 rows x 30 columns]

Observations: One notable observation is that the mean radius of the breast masses in the dataset is approximately 14.13 units, with a range spanning from 6.98 to 28.11. This suggests significant variation in the size of the masses, with some instances exhibiting considerably larger or smaller radii.

Another intriguing finding is related to the mean area of the breast masses, which is approximately

654.89 square units, with a wide range from 143.50 to 2501.00. This indicates a substantial diversity in the size and extent of the masses, ranging from relatively small to significantly large areas.

Furthermore, the average smoothness of the breast masses is approximately 0.096, with a minimum value of 0.0526 and a maximum value of 0.1634. This finding suggests that the texture and surface characteristics of the masses can vary considerably, with some exhibiting smoother surfaces and others displaying rougher textures.

Additionally, the mean concavity of the breast masses is approximately 0.0888, with a maximum value of 0.4268. This indicates that while some masses exhibit minimal concave regions, others can have significant concavity, potentially indicating more complex and irregular shapes.

Lastly, the mean fractal dimension of the breast masses is approximately 0.0628, with values ranging from 0.04996 to 0.09744. This finding suggests that the complexity and intricacy of the masses' geometric patterns can vary, with some instances displaying more intricate and self-similar structures.

These interesting findings highlight the diversity and variability within the dataset, emphasizing the importance of considering multiple features when analyzing and predicting breast cancer. The variations in size, area, texture, concavity, and fractal dimension provide valuable insights for further exploration and potential correlations with the diagnosis of breast cancer.

Distribution of Diagnosis Feature

```
[305]: print('Count of Malignant or Benign cells in diagnosis')
MyData['diagnosis'].value_counts()
```

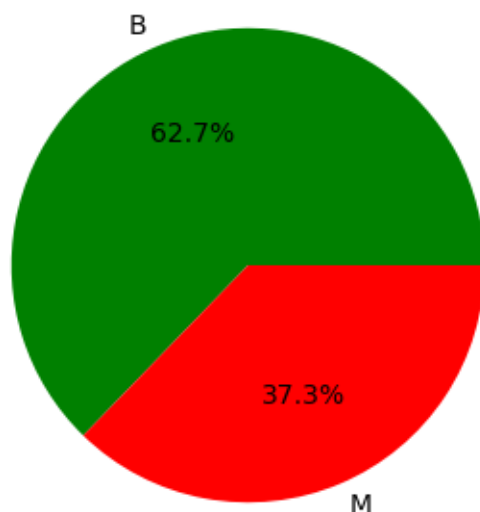
Count of Malignant or Benign cells in diagnosis

```
[305]: B    357
      M    212
      Name: diagnosis, dtype: int64
```

```
[306]: # Get the value counts of the "diagnosis" column
diagnosis_counts = MyData["diagnosis"].value_counts()

# Plotting the pie chart
plt.figure(figsize=(4, 4))
plt.pie(diagnosis_counts, labels=diagnosis_counts.index, autopct='%1.1f%%',
        colors=["green", "red"])
plt.title("Distribution of Diagnosis")
plt.show()
```

Distribution of Diagnosis



There are 62.7% Benign (Noncancerous) cells and 37.3% Malignant (Cancerous) cells in the dataset.

Histogram for the Numerical Features

```
[307]: # Defining the number of rows and columns for the grid layout
nrows = 5
ncolumns = (len(Data.columns) - 1) // nrows

# Creating a grid layout for the feature distribution plots
fig, axes = plt.subplots(nrows, ncolumns, figsize=(15, 10))

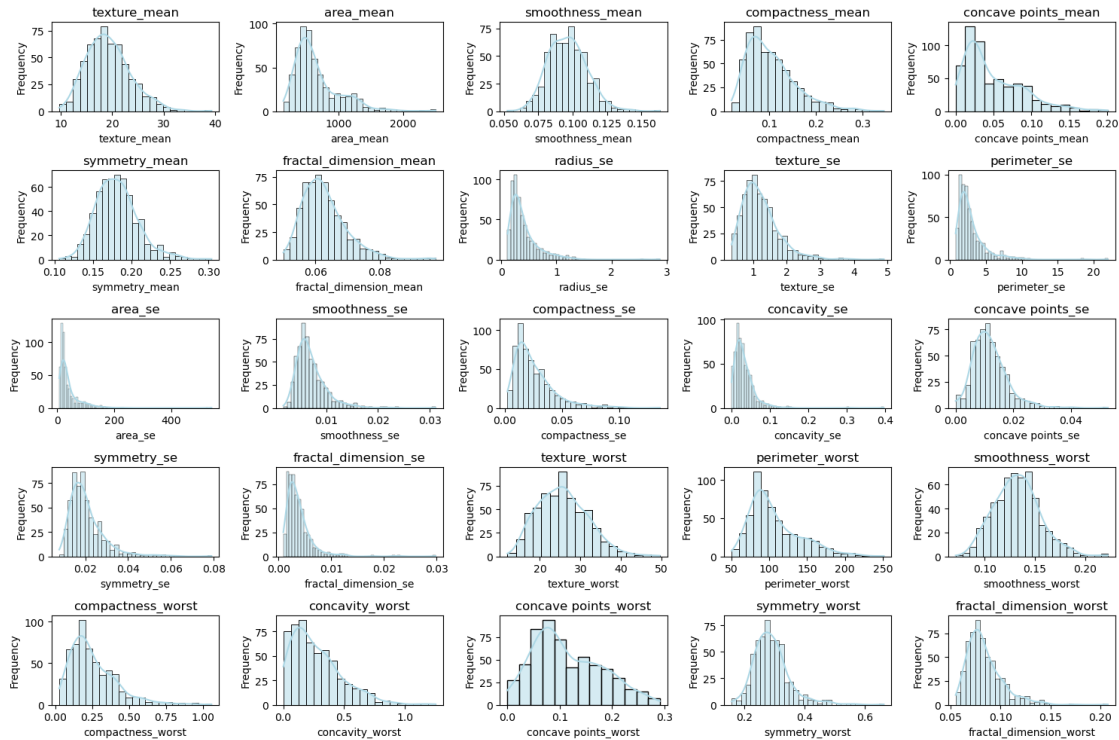
# Flattening the axes array for easy iteration
axes = axes.flatten()

# Iterating over each feature column and create a distribution plot in the
↳corresponding axis
for a, column in enumerate(Data.columns[1:]):
    sns.histplot(data=Data, x=column, kde=True, ax=axes[a], color='lightblue')
    axes[a].set_title(column)
    axes[a].set_xlabel(column)
    axes[a].set_ylabel("Frequency")

# Hide any unused subplots
if len(Data.columns) < len(axes):
    for q in range(len(Data.columns), len(axes)):
        axes[q].axis("off")
```

```
# Adjusting the layout spacing
plt.tight_layout()

# Displaying the plots
plt.show()
```



Interpretation: The data is normally distributed for most of the columns except for radius_se, parameter_se, area_se, and compactness_se which are right skewed.

Box plots for the Means Numerical Features by Group

```
[308]: ## Defining the rows and columns for the grid layout
rows = 2
columns = 5

# Create a grid layout of box plots for each feature grouped by "diagnosis"
fig, axes = plt.subplots(rows, columns, figsize=(15, 10))

# Flatten the axes array for easy iteration
axes = axes.flatten()

# Features to be plotted
```

```

features_to_plot = ["radius_mean", "texture_mean", "perimeter_mean",
                    "area_mean", "smoothness_mean",
                    "compactness_mean", "concavity_mean", "concave_
                    points_mean", "symmetry_mean",
                    "fractal_dimension_mean"]

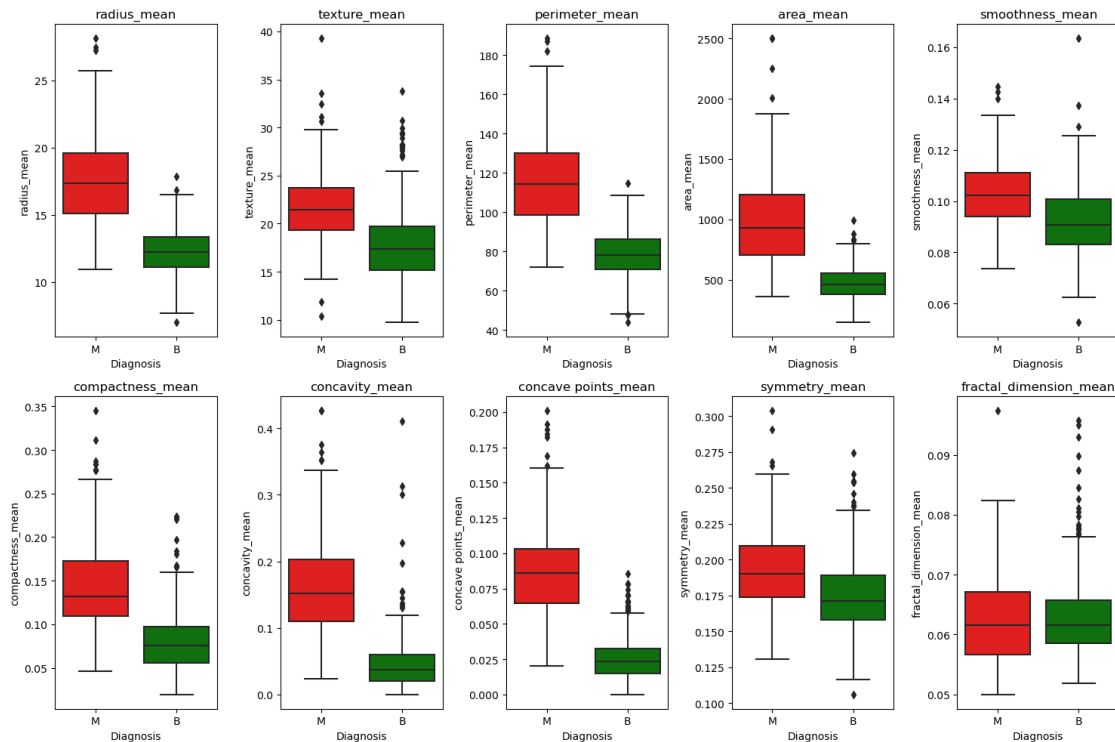
# Iterate over each feature column and create a box plot in the corresponding
axis
for a, column in enumerate(features_to_plot):
    sns.boxplot(x="diagnosis", y=column, data=MyData, ax=axes[a], palette={"M":
    "red", "B": "green"})
    axes[a].set_title(column)
    axes[a].set_xlabel("Diagnosis")
    axes[a].set_ylabel(column)

# Hide any unused subplots
if len(features_to_plot) < len(axes):
    for b in range(len(features_to_plot), len(axes)):
        axes[b].axis("off")

# Adjusting the layout spacing
plt.tight_layout()

# Displaying the plots
plt.show()

```



Findings:

- Radius of the Malignant tumours (Cancerous cells) is larger than the Benign (Noncancerous cells) this suggests that cancerous cells tend to grow and spread, resulting in tumors with a larger overall size. In contrast, benign cells typically exhibit a smaller radius, indicating a more confined and localized growth pattern
- Tissue average of the Malignant (Cancerous cells) is larger than the Benign (Noncancerous cells) this implies that cancerous cells tend to occupy a greater extent of the tissue, potentially indicating more aggressive and invasive behavior. Benign cells, on the other hand, occupy a relatively smaller area within the tissue.
- Perimeter thickness of the Malignant (Cancerous cells) is larger than the Benign (Noncancerous cells) this indicates that the boundary or edges of cancerous tumors are typically more irregular and spread out, while benign cells tend to have a smoother and more well-defined perimeter.
- The area occupied by Malignant (Cancerous cells) is larger than Benign (Noncancerous cells).
- Compactness_mean for Malignant (Cancerous cells) is higher which indicates that the points on the mass are more closely packed together as compared to the Benign (Noncancerous cells).
- Concavity mean for Malignant (Cancerous cells) is larger than the Benign (Noncancerous cells) which indicates a greater severity of concave regions within the mass of Malignant cells.
- Concave point mean is much higher for the Malignant (Cancerous cells) indicating that there are more concave regions as compared to Benign (Noncancerous cells).
- The outliers have been ignored as they do not make any difference in the prediction.

CORRELATION ANALYSIS

```
[309]: Corr_Mat= MyData.corr()
```

```
[310]: fig, ab = plt.subplots(figsize = (18, 14))
ab = sns.heatmap(Corr_Mat,
                  annot = True,
                  cbar = False)
```

	radius_mean	-1	0.32	1	0.99	0.17	0.51	0.68	0.82	0.15	-0.31	0.68	-0.097	0.67	0.74	-0.22	0.21	0.19	0.38	-0.1	-0.043	0.97	0.3	0.97	0.94	0.12	0.41	0.53	0.74	0.16	0.0071	
	texture_mean	0.32	1	0.33	0.32	0.023	0.24	0.3	0.29	0.071	-0.076	0.28	0.39	0.28	0.26	0.0066	0.19	0.14	0.16	0.0091	0.054	0.35	0.91	0.36	0.34	0.078	0.28	0.3	0.3	0.11	0.12	
	perimeter_mean	-1	0.33	1	0.99	0.21	0.56	0.72	0.85	0.18	-0.26	0.69	-0.087	0.69	0.74	-0.2	0.25	0.23	0.41	-0.082	-0.0059	0.97	0.3	0.97	0.94	0.15	0.46	0.56	0.77	0.19	0.051	
	area_mean	-0.99	0.32	0.99	1	0.18	0.5	0.69	0.82	0.15	-0.28	0.73	-0.066	0.73	0.8	-0.17	0.21	0.21	0.37	-0.072	-0.02	0.96	0.29	0.96	0.96	0.12	0.39	0.51	0.72	0.14	0.0037	
	smoothness_mean	0.17	-0.023	0.21	0.18	1	0.66	0.52	0.55	0.56	0.58	0.3	0.068	0.3	0.25	0.33	0.32	0.25	0.38	0.2	0.28	0.21	0.036	0.24	0.21	0.81	0.47	0.43	0.5	0.39	0.5	
	compactness_mean	0.51	0.24	0.56	0.5	0.66	1	0.88	0.83	0.6	0.57	0.5	0.046	0.55	0.46	0.14	0.74	0.57	0.64	0.23	0.51	0.54	0.25	0.59	0.51	0.57	0.87	0.82	0.82	0.51	0.69	
	concavity_mean	0.68	0.3	0.72	0.69	0.52	0.88	1	0.92	0.5	0.34	0.63	0.076	0.66	0.62	0.099	0.67	0.69	0.68	0.18	0.45	0.68	0.3	0.73	0.68	0.45	0.75	0.88	0.86	0.41	0.078	
	concave points_mean	0.82	0.29	0.85	0.82	0.55	0.83	0.92	1	0.46	0.17	0.021	0.71	0.69	0.028	0.49	0.44	0.62	0.095	0.26	0.83	0.29	0.86	0.81	0.45	0.67	0.75	0.91	0.38	0.37		
	symmetry_mean	0.15	0.071	0.18	0.15	0.56	0.6	0.5	0.46	1	0.48	0.3	0.13	0.31	0.22	0.19	0.42	0.34	0.34	0.69	0.45	0.33	0.19	0.091	0.22	0.18	0.43	0.47	0.43	0.43	0.7	0.44
	fractal_dimension_mean	-0.31	-0.076	-0.26	-0.28	0.58	0.57	0.34	0.17	0.48	1	0.0011	0.16	0.04	-0.09	0.4	0.56	0.45	0.34	0.35	0.69	-0.25	-0.051	-0.21	-0.23	0.5	0.46	0.35	0.18	0.33	0.77	
	radius_se	0.68	0.28	0.69	0.73	0.3	0.5	0.63	0.7	0.3	0.0001	1	0.21	0.97	0.95	0.16	0.36	0.33	0.51	0.24	0.23	0.72	0.19	0.72	0.75	0.14	0.29	0.38	0.53	0.095	0.05	
	texture_se	-0.097	0.39	-0.087	-0.066	0.068	0.046	0.076	0.021	0.13	0.16	0.21	1	0.22	0.11	0.4	0.23	0.19	0.23	0.41	0.28	-0.11	0.41	-0.1	-0.083	-0.074	-0.092	-0.069	-0.12	-0.13	-0.046	
	perimeter_se	0.67	0.28	0.69	0.73	0.3	0.55	0.66	0.71	0.31	0.04	0.97	0.22	1	0.94	0.15	0.42	0.36	0.56	0.27	0.24	0.7	0.2	0.72	0.73	0.13	0.34	0.42	0.55	0.11	0.085	
	area_se	0.74	0.26	0.74	0.8	0.25	0.46	0.62	0.69	0.22	-0.09	0.95	0.11	0.94	1	0.075	0.28	0.27	0.42	0.13	0.76	0.2	0.76	0.81	0.13	0.28	0.39	0.54	0.074	0.018		
	smoothness_se	-0.22	0.0066	-0.2	-0.17	0.33	0.14	0.099	0.028	0.19	0.4	0.16	0.4	0.15	0.075	1	0.34	0.27	0.33	0.41	0.43	-0.23	-0.075	-0.22	-0.18	0.31	-0.056	-0.058	-0.1	-0.11	0.1	
	compactness_se	0.21	0.19	0.25	0.21	0.32	0.74	0.67	0.49	0.42	0.56	0.36	0.23	0.42	0.28	0.34	1	0.8	0.74	0.39	0.8	0.2	0.14	0.26	0.2	0.23	0.68	0.64	0.48	0.28	0.59	
	concavity_se	0.19	0.14	0.23	0.21	0.25	0.57	0.69	0.44	0.34	0.45	0.33	0.19	0.36	0.27	0.27	0.8	1	0.77	0.31	0.73	0.19	0.1	0.23	0.19	0.77	0.48	0.66	0.44	0.2	0.44	
	concave points_se	0.38	0.16	0.41	0.37	0.38	0.64	0.68	0.62	0.39	0.34	0.51	0.23	0.56	0.42	0.33	0.74	0.77	1	0.31	0.61	0.36	0.087	0.39	0.34	0.22	0.45	0.55	0.6	0.14	0.31	
	symmetry_se	-0.1	0.0091	-0.082	-0.072	0.2	0.23	0.18	0.095	0.45	0.35	0.24	0.41	0.27	0.13	0.41	0.39	0.31	0.31	1	0.37	-0.13	-0.077	-0.1	-0.11	-0.013	0.06	0.037	-0.03	0.39	0.078	
	fractal_dimension_se	-0.043	0.054	-0.0055	-0.02	0.28	0.51	0.45	0.26	0.33	0.69	0.23	0.28	0.24	0.13	0.43	0.8	0.73	0.61	0.37	1	-0.037	-0.0032	-0.001	-0.023	0.17	0.39	0.38	0.22	0.11	0.59	
	radius_worst	0.97	0.35	0.97	0.96	0.21	0.54	0.69	0.83	0.19	-0.25	0.72	-0.11	0.7	0.76	-0.23	0.2	0.19	0.36	-0.13	-0.037	1	0.36	0.99	0.98	0.22	0.48	0.57	0.79	0.24	0.093	
	texture_worst	0.3	0.91	0.3	0.29	0.036	0.25	0.3	0.29	0.091	-0.051	0.19	0.41	0.2	0.2	-0.075	0.14	0.1	0.087	-0.077	-0.032	0.36	1	0.37	0.35	0.23	0.36	0.37	0.36	0.23	0.22	
	perimeter_worst	0.97	0.36	0.97	0.96	0.24	0.59	0.73	0.86	0.22	-0.21	0.72	-0.1	0.72	0.76	-0.22	0.26	0.23	0.39	-0.1	-0.001	0.99	0.37	1	0.98	0.24	0.53	0.62	0.82	0.27	0.14	
	area_worst	0.94	0.34	0.94	0.96	0.21	0.51	0.68	0.81	0.18	-0.23	0.75	-0.083	0.73	0.81	-0.18	0.2	0.19	0.34	-0.11	-0.023	0.98	0.35	0.98	1	0.21	0.44	0.54	0.75	0.21	0.08	
	smoothness_worst	0.12	0.078	0.15	0.12	0.81	0.57	0.45	0.45	0.43	0.5	0.14	-0.074	0.13	0.13	0.31	0.23	0.17	0.22	-0.013	0.17	0.22	0.23	0.24	0.21	1	0.57	0.52	0.55	0.49	0.62	
	compactness_worst	0.41	0.28	0.46	0.39	0.47	0.87	0.75	0.67	0.47	0.46	0.29	-0.092	0.34	0.28	-0.056	0.68	0.48	0.45	0.06	0.39	0.48	0.36	0.53	0.44	0.57	1	0.89	0.8	0.61	0.81	
	concavity_worst	0.53	0.3	0.56	0.51	0.43	0.82	0.88	0.75	0.43	0.35	0.38	-0.069	0.42	0.39	-0.058	0.64	0.66	0.55	0.037	0.38	0.57	0.37	0.62	0.54	0.52	0.89	1	0.86	0.53	0.69	
	concave points_worst	0.74	0.3	0.77	0.72	0.5	0.82	0.86	0.91	0.43	0.18	0.53	-0.12	0.55	0.54	-0.1	0.48	0.44	0.6	-0.03	0.22	0.79	0.36	0.82	0.75	0.55	0.8	0.86	1	0.5	0.51	
	symmetry_worst	0.16	0.11	0.19	0.14	0.39	0.51	0.41	0.38	0.7	0.33	0.095	-0.13	0.11	0.074	-0.11	0.28	0.2	0.14	0.39	0.11	0.24	0.23	0.27	0.21	0.49	0.61	0.53	0.5	1	0.54	
	fractal_dimension_worst	-0.0071	0.12	0.051	0.0037	0.5	0.69	0.51	0.37	0.44	0.77	0.05	-0.046	0.085	0.018	0.1	0.59	0.44	0.31	0.078	0.59	0.093	0.22	0.14	0.08	0.62	0.81	0.69	0.51	0.54	1	
	radius_mean												radius_se																			
	texture_mean												texture_se																			
	perimeter_mean												perimeter_se																			
	area_mean												area_se																			
	smoothness_mean												smoothness_se																			
	compactness_mean												compactness_se																			
	concavity_mean												concavity_se																			
	concave points_mean												concave points_se																			
	symmetry_mean												symmetry_se																			
	fractal_dimension_mean												fractal_dimension_se																			
	radius_worst												radius_se																			
	texture_worst												texture_se																			
	perimeter_worst												perimeter_se																			
	area_worst												area_se																			
	smoothness_worst												smoothness_se																			
	compactness_worst												compactness_se																			
	concavity_worst												concavity_se																			
	concave points_worst												concave points_se																			
	symmetry_worst												symmetry_se																			
	fractal_dimension_worst												fractal_dimension_se																			

Positive correlations:

- A significant positive correlation with a coefficient of 0.997855 was found between the variables “radius_mean” and “perimeter_mean” this is because an increase in the radius of a tumor (radius_mean) would generally result in an increase in its perimeter (perimeter_mean). This direct relationship leads to a strong positive correlation.
- The variables “radius_mean” and “area_mean” displayed a strong positive correlation, with a coefficient of 0.987357. This is because “area_mean” and “perimeter_mean” both depend on the size and shape of the tumor. As the tumor’s size increases, both its area and perimeter are likely to increase, leading to a strong positive correlation between the two variables.
- A robust positive correlation was observed between “radius_mean” and “radius_worst,” with a coefficient of 0.969539 this is because both variables capture the overall size or extent of the tumor. so, it is expected that as the mean radius of a tumor increases, the worst (largest) radius measurement would also tend to increase.
- The variables “area_mean” and “perimeter_mean” showed a strong positive correlation, with a coefficient of 0.986507. This shows that when area of tumor increases the perimeter will also increase which is quite obvious.
- A substantial positive correlation was detected between “area_mean” and “area_worst,” with

a coefficient of 0.959213.

- A strong positive correlation of 0.883121 was found between “compactness_mean” and “concavity_mean”. This tells us that tumors that exhibit higher compactness tend to have more concave regions, resulting in a positive correlation between these variables. This correlation could indicate that tumors with a higher degree of compactness are more likely to have a greater level of concavity.

Negative correlation:

- A moderate negative correlation, with a coefficient of -0.311631, was observed between “radius_mean” and “fractal_dimension_mean”. A negative correlation between these variables could indicate that tumors with larger cell radii tend to have less irregular or more uniform shapes, resulting in a lower fractal dimension.

```
[311]: #Dropping column since they are highly correlated with eachother
```

```
columns_to_drop = ['radius_mean', 'perimeter_mean', 'area_worst',  
                  ↪ 'radius_worst', 'concavity_mean']  
MyData = MyData.drop(columns=columns_to_drop)
```

0.2 METHODS

The data problem in this study was to accurately classify breast cancer cases based on a dataset containing various tumor characteristics. The goal was to develop a predictive model that could effectively distinguish between benign and malignant tumors.

To analyze breast cancer classification, this study employed several machine learning algorithms, including Random Forest, Decision Tree, Logistic Regression, KNeighbors Classifier, SVM, and Naive Bayes. These algorithms were selected due to their effectiveness in binary classification tasks. The dataset was preprocessed by label encoding the target variable, “diagnosis,” and performing feature scaling using the StandardScaler function to ensure compatibility across the algorithms. The dataset was then split into training and testing sets with a 70:30 ratio. Each algorithm was trained on the training set and evaluated on the testing set using accuracy, precision, and recall metrics. Based on the evaluation results, the Logistic Regression algorithm demonstrated the highest performance in terms of accuracy and recall, making it the algorithm of choice for further analysis. The feature importance of the logistic regression model was assessed by examining the absolute coefficients, with higher magnitudes indicating stronger influences on the breast cancer prediction. Additionally, the performance of the selected model was visualized using a confusion matrix to gain insights into its predictive performance.

Label Encoding for Target Variable

```
[312]: from sklearn.preprocessing import LabelEncoder  
       # Create an instance of the LabelEncoder  
       LE = LabelEncoder()  
  
       # Use the LabelEncoder to transform the 'diagnosis' column of the DataFrame  
       MyData['diagnosis'] = LE.fit_transform(MyData.diagnosis)
```

```
# Display the head of the modified DataFrame
MyData.head()
```

```
[312]:
```

	diagnosis	texture_mean	area_mean	smoothness_mean	compactness_mean	\
0	1	10.38	1001.0	0.11840	0.27760	
1	1	17.77	1326.0	0.08474	0.07864	
2	1	21.25	1203.0	0.10960	0.15990	
3	1	20.38	386.1	0.14250	0.28390	
4	1	14.34	1297.0	0.10030	0.13280	

	concave	points_mean	symmetry_mean	fractal_dimension_mean	radius_se	\
0		0.14710	0.2419	0.07871	1.0950	
1		0.07017	0.1812	0.05667	0.5435	
2		0.12790	0.2069	0.05999	0.7456	
3		0.10520	0.2597	0.09744	0.4956	
4		0.10430	0.1809	0.05883	0.7572	

	texture_se	...	symmetry_se	fractal_dimension_se	texture_worst	\
0	0.9053	...	0.03003	0.006193	17.33	
1	0.7339	...	0.01389	0.003532	23.41	
2	0.7869	...	0.02250	0.004571	25.53	
3	1.1560	...	0.05963	0.009208	26.50	
4	0.7813	...	0.01756	0.005115	16.67	

	perimeter_worst	smoothness_worst	compactness_worst	concavity_worst	\
0	184.60	0.1622	0.6656	0.7119	
1	158.80	0.1238	0.1866	0.2416	
2	152.50	0.1444	0.4245	0.4504	
3	98.87	0.2098	0.8663	0.6869	
4	152.20	0.1374	0.2050	0.4000	

	concave	points_worst	symmetry_worst	fractal_dimension_worst
0		0.2654	0.4601	0.11890
1		0.1860	0.2750	0.08902
2		0.2430	0.3613	0.08758
3		0.2575	0.6638	0.17300
4		0.1625	0.2364	0.07678

[5 rows x 26 columns]

Splitting the data into target and predictor variables

```
[313]: predictors = MyData.drop(['diagnosis'], axis=1)
target = MyData.diagnosis
```

Split the data into training and testing set


```
[314]: from sklearn.model_selection import train_test_split as split
X_train, X_test, Y_train, Y_test = split(predictors, target, test_size=0.3,
↳ random_state=0)
```

Feature Scaling for numerical variables

```
[315]: from sklearn.preprocessing import StandardScaler
# Create an instance of the StandardScaler
sc = StandardScaler()

# Fit the scaler on the training data and transform it
X_train = sc.fit_transform(X_train)

# Transform the test data using the fitted scaler
X_test = sc.transform(X_test)
```

0.2.1 MODELLING

0.2.2 Applying Supervised Learning Algorithms

```
[316]: from sklearn.metrics import accuracy_score, confusion_matrix, precision_score,
↳ recall_score
classifiers = {
    "Random Forest": RandomForestClassifier(),
    "Decision Tree": DecisionTreeClassifier(),
    "Logistic Regression": LogisticRegression(),
    "KNeighbors Classifier": KNeighborsClassifier(),
    "SVM": SVC(),
    "Naive Bayes": GaussianNB()
}

# Create lists to store the results
classifier_list = []
accuracy_list = []
precision_list = []
recall_list = []

for clf_name, clf in classifiers.items():
    # Fitting on training set
    clf.fit(X_train, Y_train)

    # predictions on the training set
    Y_train_prediction = clf.predict(X_train)

    # Calculating accuracy, precision, and recall on the training data
    accuracy_train = accuracy_score(Y_train, Y_train_prediction)
    precision_train = precision_score(Y_train, Y_train_prediction)
```

```

recall_train = recall_score(Y_train, Y_train_prediction)

# for predictions on the test set
Y_test_prediction = clf.predict(X_test)

# Calculating the accuracy, precision, and recall on the test set
accuracy_test = accuracy_score(Y_test, Y_test_prediction)
precision_test = precision_score(Y_test, Y_test_prediction)
recall_test = recall_score(Y_test, Y_test_prediction)

# Add the results to the lists
classifier_list.append(clf_name)
accuracy_list.append(accuracy_test)
precision_list.append(precision_test)
recall_list.append(recall_test)

# Create a DataFrame from the lists
data = {
    "Classifier": classifier_list,
    "Accuracy": accuracy_list,
    "Precision": precision_list,
    "Recall": recall_list
}
my_result = pd.DataFrame(data)

# Sorting by Accuracy
my_result = df.sort_values(by="Accuracy", ascending=False)

# Printing the sorted DataFrame
print(my_result)

```

	Classifier	Accuracy	Precision	Recall
2	Logistic Regression	0.970760	0.967742	0.952381
4	SVM	0.970760	0.967742	0.952381
3	KNeighbors Classifier	0.953216	0.966102	0.904762
0	Random Forest	0.947368	0.935484	0.920635
1	Decision Tree	0.929825	0.849315	0.984127
5	Naive Bayes	0.894737	0.857143	0.857143

Comparison Result:

Since Recall is the ratio of true positive predictions to the total actual positives. It represents the model's ability to correctly identify all positive cases in the dataset. In breast cancer prediction, recall indicate the proportion of correctly identified malignant cases out of all actual malignant cases which is important because the cost of false negatives (incorrectly predicting in our case as benign) is high. Since Logistic Regression and SVM both have same highest Recall and accuracy, we will select one of them for further exploration. Let us select Logistic Regression Model.

0.2.3 LOGISTIC REGRESSION MODEL

List of important features

```
[317]: import matplotlib.pyplot as plt

# Creating an instance of the LogisticRegression model
logisticreg = LogisticRegression()

# Fitting model on train set
logisticreg.fit(X_train, Y_train)

# Making predictions on the train set
Y_train_prediction = logisticreg.predict(X_train)

# Calculating the accuracy on the training set
accuracy_train = accuracy_score(Y_train, Y_train_prediction)

# Making predictions on test set
Y_test_prediction = logisticreg.predict(X_test)

# Getting feature importance from logistic regression model
feature_importance = logisticreg.coef_[0]

# Sorting the features and coefficients in descending order of absolute
# coefficients
sort_features = sorted(zip(X.columns, feature_importance), key=lambda x:
# abs(x[1]), reverse=True)

# Printing the important features with coefficients
print("Important Features:")
for labels, coef in sort_features:
    print(f"{labels}: {coef}")
```

Important Features:

```
radius_se: 1.5001780772439868
perimeter_worst: 1.3345035510697556
concave points_worst: 1.1969150315085257
area_se: 1.161434256038746
concave points_mean: 1.0875217243611088
concavity_worst: 1.039052179201762
area_mean: 1.030286728652244
perimeter_se: 1.0012419908392547
texture_worst: 0.8897330315699753
compactness_se: -0.8864372679131869
fractal_dimension_se: -0.8377845122500643
texture_mean: 0.58393322362725
fractal_dimension_worst: 0.5666095629050769
smoothness_worst: 0.5435451884256961
```

```
symmetry_worst: 0.45703657552294213
symmetry_mean: 0.3191954083837519
smoothness_se: -0.2865823733438372
concave_points_se: 0.27377156800404595
fractal_dimension_mean: -0.2633949867058265
compactness_mean: -0.25327142510849937
concavity_se: 0.2490336637866851
texture_se: -0.1989350836502589
compactness_worst: -0.1719955341479354
smoothness_mean: 0.12440945921866486
symmetry_se: -0.11595649472402751
```

Findings:

The coefficients of the logistic regression model reveal the impact of each feature on the prediction of breast cancer classification. These coefficients represent the estimated change in the log-odds of having breast cancer for a one-unit increase in the corresponding feature, while holding other features constant.

By examining the list of coefficients, we can draw certain conclusions. Positive coefficients (such as `radius_se`, `perimeter_worst`, and `concave_points_worst`) indicate that an increase in these features is associated with a higher likelihood of having breast cancer. On the other hand, negative coefficients (such as `texture_mean`, `smoothness_mean`, and `symmetry_se`) suggest that an increase in these features is linked to a lower likelihood of having breast cancer. The magnitude of the coefficients is also important. Larger magnitude coefficients imply a stronger influence on the prediction of breast cancer.

Since, `radius_se` has the highest magnitude and is positive so it is the most important feature for the prediction of breast cancer when using Logistic regression. It is a feature that quantifies the standard error of the mean of distances from the center to points on the perimeter of a breast mass.

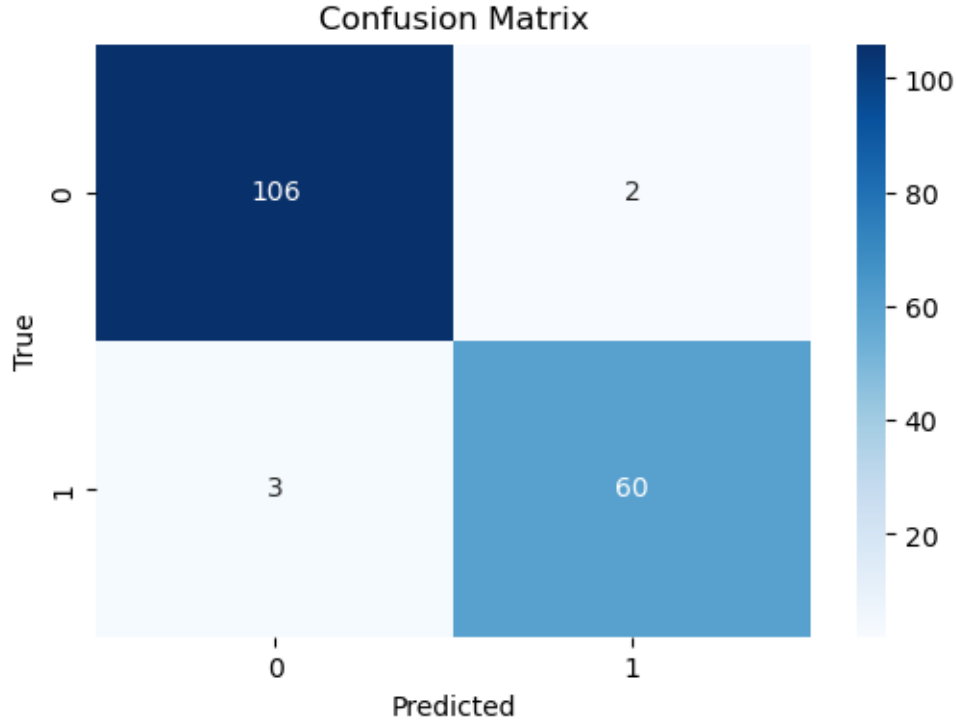
Plotting Confusion Matrix

```
[318]: # Generate the confusion matrix
confusion_mat = confusion_matrix(Y_test, Y_test_pred)

# Plotting the confusion matrix
plt.figure(figsize=(6, 4))
sns.heatmap(confusion_mat, annot=True, fmt="d", cmap="Blues")

# Setting labels
plt.xlabel("Predicted")
plt.ylabel("True")
plt.title("Confusion Matrix")

# Displaying the plot
plt.show()
```



- The top-left cell (106) represents the number of true negatives (TN). It indicates that there are 106 instances that were correctly predicted as the negative class.
- The top-right cell (2) represents the number of false positives (FP). It indicates that there are 2 instances that were incorrectly predicted as the positive class when they were actually negative.
- The bottom-left cell (3) represents the number of false negatives (FN). It indicates that there are 3 instances that were incorrectly predicted as the negative class when they were actually positive.
- The bottom-right cell (60) represents the number of true positives (TP). It indicates that there are 60 instances that were correctly predicted as the positive class.

0.3 RESULTS

The study evaluated the performance of several machine learning algorithms for the classification of breast cancer cases based on tumor characteristics. Among the models tested, Logistic Regression emerged as the top performer, exhibiting the highest accuracy and recall scores on both the training and test datasets. The Logistic Regression model achieved an impressive accuracy rate of 97%, indicating its capability to accurately predict breast cancer cases. Moreover, the recall score, which measures the model's ability to correctly identify positive cases, was also high at 95.2%, underscoring the model's effectiveness in detecting malignant tumors.

Additionally, the feature importance analysis revealed that the “radius_se” feature exhibited the highest magnitude and positive coefficient in the logistic regression model. This finding suggests

that the variability in tumor radius size played a significant role in predicting breast cancer cases. Other influential features such as “smoothness_se,” “texture_se,” and “symmetry_se” also contributed to the overall predictive power of the model.

To provide a comprehensive overview of the logistic regression model’s performance, a confusion matrix was employed for visualization. The confusion matrix displayed a relatively low number of false positives and false negatives, further supporting the model’s accuracy and reliability.

In summary, the results underscore the effectiveness of the Logistic Regression algorithm in accurately classifying breast cancer cases based on tumor characteristics. These findings indicate the potential of the selected model as a valuable tool to aid medical professionals in the early detection and diagnosis of breast cancer, ultimately leading to improved patient outcomes and timely interventions.

0.4 CONCLUSION

The Project aimed to develop an accurate and efficient breast cancer prediction model using supervised learning algorithms. The dataset consisted of 569 instances with various features related to breast cancer diagnosis. The dataset was preprocessed by label encoding the target variable and performing feature scaling.

Descriptive analysis of the dataset revealed interesting findings, such as significant variations in the size, area, texture, concavity, and fractal dimension of the breast masses. These variations provided valuable insights into the characteristics of malignant and benign tumors, emphasizing the importance of considering multiple features when diagnosing breast cancer.

Through a comparative analysis of different algorithms, including logistic regression, support vector machines, decision trees, random forests, SVM, and Naive Bayes, the logistic regression algorithm demonstrated the highest performance in terms of accuracy and recall. Hence, it was chosen as the algorithm of choice for further analysis.

The logistic regression model highlighted the importance of the “radius_se” feature, which represents the standard error of the mean of distances from the center to points on the perimeter. This feature had the highest magnitude coefficient and was identified as the most influential feature for breast cancer prediction using logistic regression.

Evaluation of the logistic regression model using a confusion matrix showed good performance in correctly predicting both malignant and benign cases. The model achieved a high number of true negatives and true positives, indicating its ability to accurately classify both types of cases. However, there were a small number of false positives and false negatives, suggesting room for further improvement in future iterations.

The End