Breast Cancer Prediction Using Classification Model

Name	Id
Md Rifat Khan	20-42704-1
Md Faysal Ahmed	20-42693-1
Al-Ruhul Amin Sabbir	20-42731-1

1. Introduction

1.1 Problem Statement

Breast cancer is a prevalent and potentially life-threatening disease that affects a significant number of women worldwide. Early detection and accurate diagnosis are crucial for effective treatment and increased survival rates. Medical professionals have traditionally relied on various diagnostic methods, including mammography and biopsy. However, with the advent of machine learning, there's an opportunity to enhance the accuracy and efficiency of breast cancer detection. Moreover, the combinations of all the model parameters are considered for simulations. The experimental results of the classifiers show that which classifier model gives better classification accuracy. The existing solution for breast cancer diagnosis relies on manual examination of mammograms, biopsies, and patient history by medical professionals. While effective, this approach can be time-consuming and subject to human error and bias. There is a need for a more automated and objective method that can assist medical practitioners in making accurate and timely diagnoses. This project aims to address this gap by developing a machine learning model that can classify breast tumors as malignant or benign based on quantifiable features. The model will augment the diagnostic process, enabling healthcare professionals to make more informed decisions and potentially leading to earlier detection and improved patient outcomes. In this project NumPy, pandas, matplotlib , seaborn , sklearn libraries are used. Data preprocessing and Exploratory data analysis is present to justify the project as more accurate one. After data analysis unnecessary variables were removed. Then dataset is splitted into training and test set. After that a model is built for each model and test our data. At last it is compared the accuracy score.

1.2 Objective

The primary objective of this project is to design, develop, and evaluate a robust machine learning classification model that enhances the accuracy and efficiency of breast cancer detection. By analyzing quantifiable features extracted from patient data, the model aims to differentiate between

malignant and benign tumors, thereby aiding medical professionals in making informed decisions and potentially facilitating early intervention. The specific objectives of this project include:

i. Dataset Collection and Preprocessing:

Gather a comprehensive dataset comprising various features associated with breast tumor characteristics, patient histories, and relevant medical information. Preprocess the dataset to handle missing values, outliers, and inconsistencies, ensuring the quality and reliability of the data used for model training.

ii. Model Development and Training:

Implement and train machine learning classification algorithms on the preprocessed dataset. Explore a range of algorithms, such as Gaussian Naive Bayes, support vector machines, and K Nearest Neighbours , to identify the most suitable approach for the given task.

iii. Performance Evaluation:

Evaluate the trained models using appropriate evaluation metrics, including accuracy.

By achieving these objectives, we aspire to provide healthcare professionals with a valuable tool that complements their expertise in breast cancer diagnosis. The project's success will be measured by the model's ability to accurately classify tumors, contribute to early detection, reduce false positives/negatives, and enhance overall patient care.

2. Methodology

2.1 Data Collection Procedure

We are loading breast cancer data using a scikit-learn load_brast_cancer class.

Number of Instances: 569

Number of Attributes: 30 numeric, predictive attributes and the class

Attribute Information:

- radius (mean of distances from center to points on the perimeter)
- texture (standard deviation of gray-scale values)
- perimeter
- area
- smoothness (local variation in radius lengths)
- compactness (perimeter^2 / area - 1.0)

- concavity (severity of concave portions of the contour)

- concave points (number of concave portions of the contour)

- symmetry

- fractal dimension ("coastline approximation" - 1)

The mean, standard error, and "worst" or largest (mean of the three worst/largest values) of these features were computed for each image, resulting in 30 features. For instance, field 0 is Mean Radius, field 10 is Radius SE, field 20 is Worst Radius.

- class:

    - WDBC-Malignant

    - WDBC-Benign

Import Libraries

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

```

Load breast cancer dataset

```
from sklearn.datasets import load_breast_cancer
cancer_dataset = load_breast_cancer()
```

```
cancer_dataset
```

Output >>> sklearn {'data': array([[1.799e+01, 1.038e+01, 1.228e+02, ..., 2.654e-01, 4.601e-01,

1.189e-01],

[2.057e+01, 1.777e+01, 1.329e+02, ..., 1.860e-01, 2.750e-01,

8.902e-02],

[1.969e+01, 2.125e+01, 1.300e+02, ..., 2.430e-01, 3.613e-01,

8.758e-02],

...,

[1.660e+01, 2.808e+01, 1.083e+02, ..., 1.418e-01, 2.218e-01,

7.820e-02],
[2.060e+01, 2.933e+01, 1.401e+02, ..., 2.650e-01, 4.087e-01,
1.240e-01],
[7.760e+00, 2.454e+01, 4.792e+01, ..., 0.000e+00, 2.871e-01,
7.039e-02]]),
'target': array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0,
0, 0, 1, 0, 1, 1, 1, 1, 1, 0, 0, 1, 0, 0, 1, 1, 1, 1, 0, 1, 0, 0,
1, 1, 1, 1, 0, 1, 0, 0, 1, 0, 1, 0, 0, 1, 1, 1, 0, 0, 1, 0, 0, 0,
1, 1, 1, 0, 1, 1, 0, 0, 1, 1, 1, 0, 0, 1, 1, 1, 1, 0, 1, 1, 0, 1,
1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 1, 0, 0, 1, 1, 1, 0, 0, 1, 0, 1, 0,
0, 1, 0, 0, 1, 1, 0, 1, 1, 0, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1,
1, 1, 0, 1, 1, 1, 1, 0, 0, 1, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 1,
1, 0, 1, 1, 0, 0, 0, 1, 0, 1, 0, 1, 1, 1, 0, 1, 1, 0, 0, 1, 0, 0,
0, 0, 1, 0, 0, 0, 1, 0, 1, 0, 1, 1, 0, 1, 0, 0, 0, 0, 1, 1, 0, 0,
1, 1, 1, 0, 1, 1, 1, 1, 1, 0, 0, 1, 1, 0, 1, 1, 0, 0, 1, 0, 1, 1,
1, 1, 0, 1, 1, 1, 1, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 1, 1, 1, 1, 1, 1, 1, 0, 1, 0, 1, 1, 0, 1, 1, 0, 1, 0, 0, 1, 1,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 0, 1, 0, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1, 0, 1, 1, 1, 1, 0, 0,
0, 1, 1, 1, 1, 0, 1, 0, 1, 0, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 0,
0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 1, 0, 0, 0, 1, 0, 0,
1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 0, 0, 1, 1,
1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0, 1, 1, 0,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 0, 0, 1, 0, 1, 1, 1, 1,
1, 0, 1, 1, 0, 1, 0, 1, 1, 0, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0,
1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1,
1, 1, 1, 0, 1, 0, 1, 1, 0, 1, 1, 1, 1, 1, 0, 0, 1, 0, 1, 0, 1, 1,
1, 1, 1, 0, 1, 1, 0, 1, 0, 1, 0, 0, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1, 0, 1, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 1]),
'frame': None,

'target_names': array(['malignant', 'benign'], dtype='<U9'),

'DESCR': '.. _breast_cancer_dataset:\n\nBreast cancer wisconsin (diagnostic) dataset\n--------------------------------------\n\n**Data Set Characteristics:**\n\n :Number of Instances: 569\n\n :Number of Attributes: 30 numeric, predictive attributes and the class\n\n :Attribute Information:\n - radius (mean of distances from center to points on the perimeter)\n - texture (standard deviation of gray-scale values)\n - perimeter\n - area\n - smoothness (local variation in radius lengths)\n - compactness (perimeter^2 / area - 1.0)\n - concavity (severity of concave portions of the contour)\n - concave points (number of concave portions of the contour)\n - symmetry\n - fractal dimension ("coastline approximation" - 1)\n\n The mean, standard error, and "worst" or largest (mean of the three\n worst/largest values) of these features were computed for each image,\n resulting in 30 features. For instance, field 0 is Mean Radius, field\n 10 is Radius SE, field 20 is Worst Radius.\n\n - class:\n - WDBC-Malignant\n - WDBC-Benign\n\n :Summary Statistics:\n\n ===================================== ====== ======\n Min Max\n ===================================== ====== ======\n radius (mean): 6.981 28.11\n texture (mean): 9.71 39.28\n perimeter (mean): 43.79 188.5\n area (mean): 143.5 2501.0\n smoothness (mean): 0.053 0.163\n compactness (mean): 0.019 0.345\n concavity (mean): 0.0 0.427\n concave points (mean): 0.0 0.201\n symmetry (mean): 0.106 0.304\n fractal dimension (mean): 0.05 0.097\n radius (standard error): 0.112 2.873\n texture (standard error): 0.36 4.885\n perimeter (standard error): 0.757 21.98\n area (standard error): 6.802 542.2\n smoothness (standard error): 0.002 0.031\n compactness (standard error): 0.002 0.135\n concavity (standard error): 0.0 0.396\n concave points (standard error): 0.0 0.053\n symmetry (standard error): 0.008 0.079\n fractal dimension (standard error): 0.001 0.03\n radius (worst): 7.93 36.04\n texture (worst): 12.02 49.54\n perimeter (worst): 50.41 251.2\n area (worst): 185.2 4254.0\n smoothness (worst): 0.071 0.223\n compactness (worst): 0.027 1.058\n concavity (worst): 0.0 1.252\n concave points (worst): 0.0 0.291\n symmetry (worst): 0.156 0.664\n fractal dimension (worst): 0.055 0.208\n ===================================== ====== ======\n\n :Missing Attribute Values: None\n\n :Class Distribution: 212 - Malignant, 357 - Benign\n\n :Creator: Dr. William H. Wolberg, W. Nick Street, Olvi L. Mangasarian\n\n :Donor: Nick Street\n\n :Date: November, 1995\n\nThis is a copy of UCI ML Breast Cancer Wisconsin (Diagnostic) datasets.\nhttps://goo.gl/U2Uwz2\n\nFeatures are computed from a digitized image of a fine needle\naspirate (FNA) of a breast mass. They describe\ncharacteristics of the cell nuclei present in the image.\n\nSeparating plane described above was obtained using\nMultisurface Method-Tree (MSM-T) [K. P. Bennett, "Decision Tree\nConstruction Via Linear Programming." Proceedings of the 4th\nMidwest Artificial Intelligence and Cognitive Science Society,\npp. 97-101, 1992], a classification method which uses linear\nprogramming to construct a decision tree. Relevant features\nwere selected using an exhaustive search in the space of 1-4\nfeatures and 1-3 separating planes.\n\nThe

actual linear program used to obtain the separating plane\nin the 3-dimensional space is that described in:\n[K. P. Bennett and O. L. Mangasarian: "Robust Linear\nProgramming Discrimination of Two Linearly Inseparable Sets",\nOptimization Methods and Software 1, 1992, 23-34].\n\nThis database is also available through the UW CS ftp server:\n\nftp ftp.cs.wisc.edu\ncd math-prog/cpo-dataset/machine-learn/WDBC/\n\n.. topic:: References\n\n - W.N. Street, W.H. Wolberg and O.L. Mangasarian. Nuclear feature extraction \n for breast tumor diagnosis. IS&T/SPIE 1993 International Symposium on \n Electronic Imaging: Science and Technology, volume 1905, pages 861-870,\n San Jose, CA, 1993.\n - O.L. Mangasarian, W.N. Street and W.H. Wolberg. Breast cancer diagnosis and \n prognosis via linear programming. Operations Research, 43(4), pages 570-577, \n July-August 1995.\n - W.H. Wolberg, W.N. Street, and O.L. Mangasarian. Machine learning techniques\n to diagnose breast cancer from fine-needle aspirates. Cancer Letters 77 (1994) \n 163-171.',

'feature_names': array(['mean radius', 'mean texture', 'mean perimeter', 'mean area',

'mean smoothness', 'mean compactness', 'mean concavity',

'mean concave points', 'mean symmetry', 'mean fractal dimension',

'radius error', 'texture error', 'perimeter error', 'area error',

'smoothness error', 'compactness error', 'concavity error',

'concave points error', 'symmetry error',

'fractal dimension error', 'worst radius', 'worst texture',

'worst perimeter', 'worst area', 'worst smoothness',

'worst compactness', 'worst concavity', 'worst concave points',

'worst symmetry', 'worst fractal dimension'], dtype='<U23'),

'filename': 'breast_cancer.csv',

'data_module': 'sklearn.datasets.data'}

```
type(cancer_dataset)
```

Output >>> sklearn.utils.Bunch

The scikit-learn store data in an object bunch like a dictionary.

```
cancer_dataset.keys()
```

Output >>> dict_keys([‘data’, ‘target’, ‘target_names’, ‘DESCR’, ‘feature_names’, ‘filename’])

```
cancer_dataset['data']
```

Output >>>

```
array([[1.799e+01, 1.038e+01, 1.228e+02, ..., 2.654e-01, 4.601e-01,
        1.189e-01],
       [2.057e+01, 1.777e+01, 1.329e+02, ..., 1.860e-01, 2.750e-01,
        8.902e-02],
       [1.969e+01, 2.125e+01, 1.300e+02, ..., 2.430e-01, 3.613e-01,
        8.758e-02],
       ...,
       [1.660e+01, 2.808e+01, 1.083e+02, ..., 1.418e-01, 2.218e-01,
        7.820e-02],
       [2.060e+01, 2.933e+01, 1.401e+02, ..., 2.650e-01, 4.087e-01,
        1.240e-01],
       [7.760e+00, 2.454e+01, 4.792e+01, ..., 0.000e+00, 2.871e-01,
        7.039e-02]])
```

These numeric values are extracted features of each cell.

```
cancer_dataset['target']
```

Output >>> array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 1, 1, 1, 1, 0, 0, 1, 0, 0, 1, 1, 1, 1,
0, 1, 0, 0, 1, 1, 1, 1, 0, 1, 0, 0, 1, 0, 1, 0, 0, 1, 1, 1, 0, 0, 1, 0, 0, 0, 1, 1, 1, 0, 1, 1, 0, 0, 1,
1, 1, 0, 0, 1, 1, 1, 1, 0, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 1, 0, 0, 1, 1, 1, 0, 0, 1, 0, 1, 0,
0, 1, 0, 0, 1, 1, 0, 1, 1, 0, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 0, 0, 1, 0, 1, 1,
0, 0, 1, 1, 0, 0, 1, 1, 1, 1, 0, 1, 1, 0, 0, 0, 1, 0, 1, 0, 1, 1, 1, 0, 1, 1, 0, 0, 1, 0, 0, 0, 0, 1, 0,
0, 0, 1, 0, 1, 0, 1, 1, 0, 1, 0, 0, 0, 0, 1, 1, 0, 0, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0, 0, 1, 1, 0, 1, 1, 0,
0, 1, 0, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1,
0, 1, 0, 1, 1, 0, 1, 1, 0, 1, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 0, 1, 0, 1, 1, 1, 1,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1, 0, 1, 1, 1, 1, 0, 0, 0, 1, 1, 1, 1, 0, 1, 0, 1, 0, 1, 1,
1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 1, 0, 0, 0, 1, 0, 0, 1, 1, 1,
1, 1, 0, 1, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 0, 0, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1,
1, 1, 0, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 0, 0, 1, 0, 1, 1, 1, 1, 1, 0, 1, 1, 0, 1, 0,
1, 1, 0, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1,
1, 1, 1, 1, 1, 0, 1, 0, 1, 1, 0, 1, 1, 1, 1, 1, 0, 0, 1, 0, 1, 0, 1, 1, 1, 1, 1, 0, 1, 1, 0, 1, 0, 1, 0,
0, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 1])

```
cancer_dataset['target_names']
```

Output >>> array(['malignant', 'benign'], dtype='<U9')

0 means malignant tumor
1 mean benign tumor

.

```
print(cancer_dataset['DESCR'])
```

Output >>>

```
print(cancer_dataset['feature_names'])
```

Output >>>

```
['mean radius' 'mean texture' 'mean perimeter' 'mean area'
 'mean smoothness' 'mean compactness' 'mean concavity'
 'mean concave points' 'mean symmetry' 'mean fractal dimension'
 'radius error' 'texture error' 'perimeter error' 'area error'
 'smoothness error' 'compactness error' 'concavity error'
 'concave points error' 'symmetry error' 'fractal dimension error'
 'worst radius' 'worst texture' 'worst perimeter' 'worst area'
 'worst smoothness' 'worst compactness' 'worst concavity'
 'worst concave points' 'worst symmetry' 'worst fractal dimension']
```

```
print(cancer_dataset['filename'])
```

Output >>> breast_cancer.csv

2.2. Data Validation Procedure

Data validation is a critical step in ensuring the quality, integrity, and suitability of your dataset for machine learning tasks. In the context of breast cancer prediction, where accurate and trustworthy data is vital, a thorough validation procedure helps identify and address potential issues before training and evaluating classification models.

```
cancer_df.info()
```

Output >>>

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 569 entries, 0 to 568
Data columns (total 31 columns):
mean radius                569 non-null float64
mean texture               569 non-null float64
mean perimeter             569 non-null float64
mean area                  569 non-null float64
mean smoothness            569 non-null float64
mean compactness           569 non-null float64
mean concavity             569 non-null float64
mean concave points        569 non-null float64
mean symmetry              569 non-null float64
mean fractal dimension     569 non-null float64
radius error               569 non-null float64
```

```
texture error                569 non-null float64
perimeter error              569 non-null float64
area error                   569 non-null float64
smoothness error             569 non-null float64
compactness error            569 non-null float64
concavity error              569 non-null float64
concave points error         569 non-null float64
symmetry error               569 non-null float64
fractal dimension error      569 non-null float64
worst radius                 569 non-null float64
worst texture                569 non-null float64
worst perimeter              569 non-null float64
worst area                   569 non-null float64
worst smoothness             569 non-null float64
worst compactness            569 non-null float64
worst concavity              569 non-null float64
worst concave points         569 non-null float64
worst symmetry               569 non-null float64
worst fractal dimension      569 non-null float64
target                       569 non-null float64
dtypes: float64(31)
memory usage: 137.9 KB

```

We have a total of non-null 569 patients' information with 31 features. All feature data types in the float. The size of the DataFrame is 137.9 KB.

Numerical distribution of data. We can know to mean, standard deviation, min, max, 25%,50% and 75% value of each feature.

```
cancer_df.describe()
```

Output >>>

	mean radius	mean texture	mean perimeter	mean area	mean smoothness	mean compactness	mean concavity	mean concave points	mean symmetry	mean fractal dimension	...	worst texture	worst perimeter	worst area	worst smoothness	worst compactness	worst concavity	worst concave points	worst symmetry	worst fractal dimension	target
count	569.000000	569.000000	569.000000	569.000000	569.000000	569.000000	569.000000	569.000000	569.000000	569.000000	...	569.000000	569.000000	569.000000	569.000000	569.000000	569.000000	569.000000	569.000000	569.000000	569.000000
mean	14.127292	19.289649	91.969033	654.889104	0.096360	0.104341	0.088799	0.048919	0.181162	0.062798	...	25.677223	107.261213	880.583128	0.132369	0.254265	0.272188	0.114606	0.290076	0.083946	0.627417
std	3.524049	4.301036	24.298981	351.914129	0.014064	0.052813	0.079720	0.038803	0.027414	0.007060	...	6.146258	33.602542	569.356993	0.022832	0.157336	0.208624	0.065732	0.061867	0.018061	0.483918
min	6.981000	9.710000	43.790000	143.500000	0.052630	0.019380	0.000000	0.000000	0.106000	0.049960	...	12.020000	50.410000	185.200000	0.071170	0.027290	0.000000	0.000000	0.156500	0.055040	0.000000
25%	11.700000	16.170000	75.170000	420.300000	0.086370	0.064920	0.029560	0.020310	0.161900	0.057700	...	21.080000	84.110000	515.300000	0.116600	0.147200	0.114500	0.064930	0.250400	0.071460	0.000000
50%	13.370000	18.840000	86.240000	551.100000	0.095870	0.092630	0.061540	0.033500	0.179200	0.061540	...	25.410000	97.660000	686.500000	0.131300	0.211900	0.226700	0.099930	0.282200	0.080040	1.000000
75%	15.780000	21.800000	104.100000	782.700000	0.105300	0.130400	0.130700	0.074000	0.195700	0.066120	...	29.720000	125.400000	1084.000000	0.146000	0.339100	0.382900	0.161400	0.317900	0.092080	1.000000
max	28.110000	39.280000	188.500000	2501.000000	0.163400	0.345400	0.426800	0.201200	0.304000	0.097440	...	49.540000	251.200000	4254.000000	0.222600	1.058000	1.252000	0.291000	0.663800	0.207500	1.000000

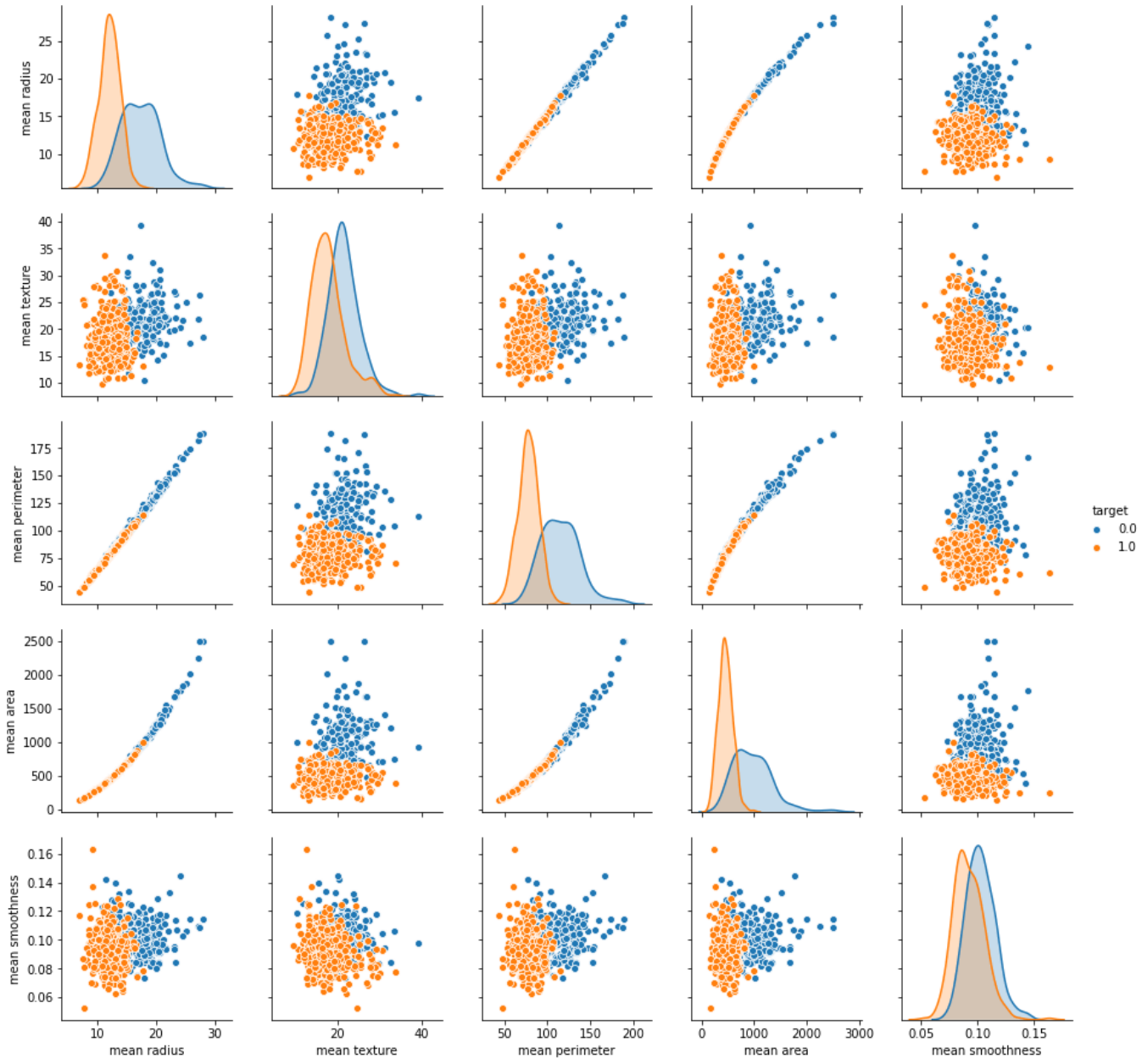
8 rows × 31 columns

```
2    sns.pairplot(cancer_df, hue = 'target')
```

Output >>>

```
sns.pairplot(cancer_df, hue = 'target',
                vars = ['mean radius', 'mean texture', 'mean perimeter', 'mean area', 'mea
```

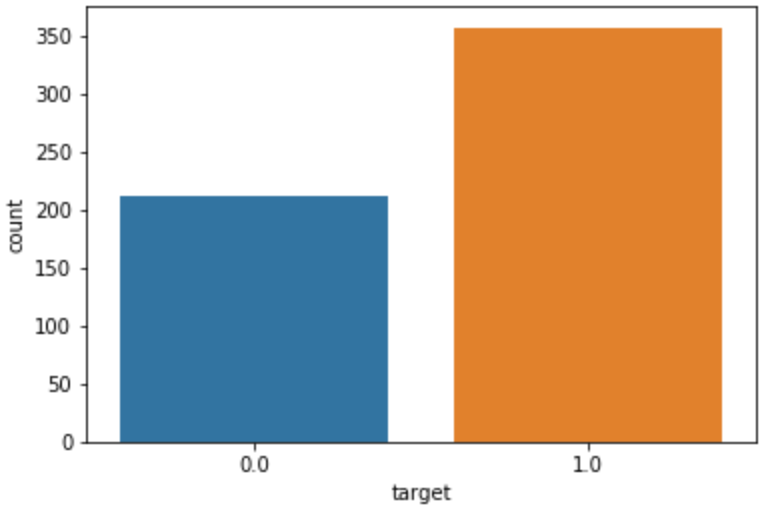
Output >>>



The pair plot showing malignant and benign tumor data distributed in two classes. It is easy to differentiate in the pair plot.

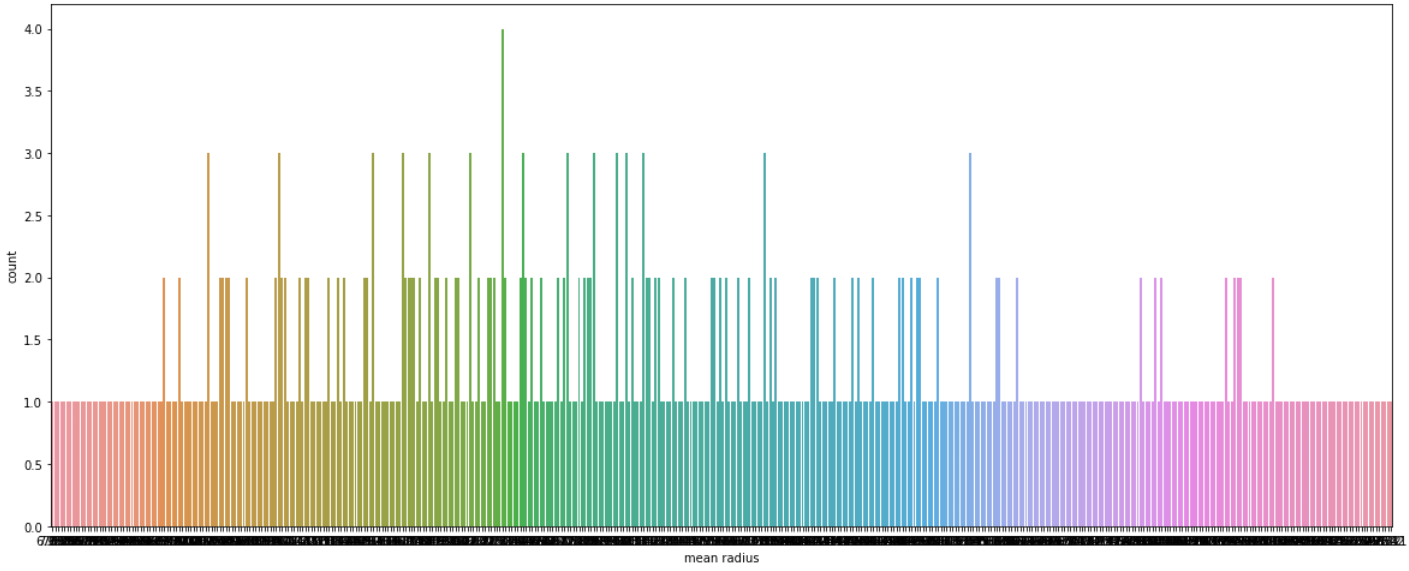
```
sns.countplot(cancer_df['target'])
```

Output >>>



In the below counterplot max samples mean radius is equal to 1.

```
plt.figure(figsize = (20,8))
sns.countplot(cancer_df['mean radius'])
```

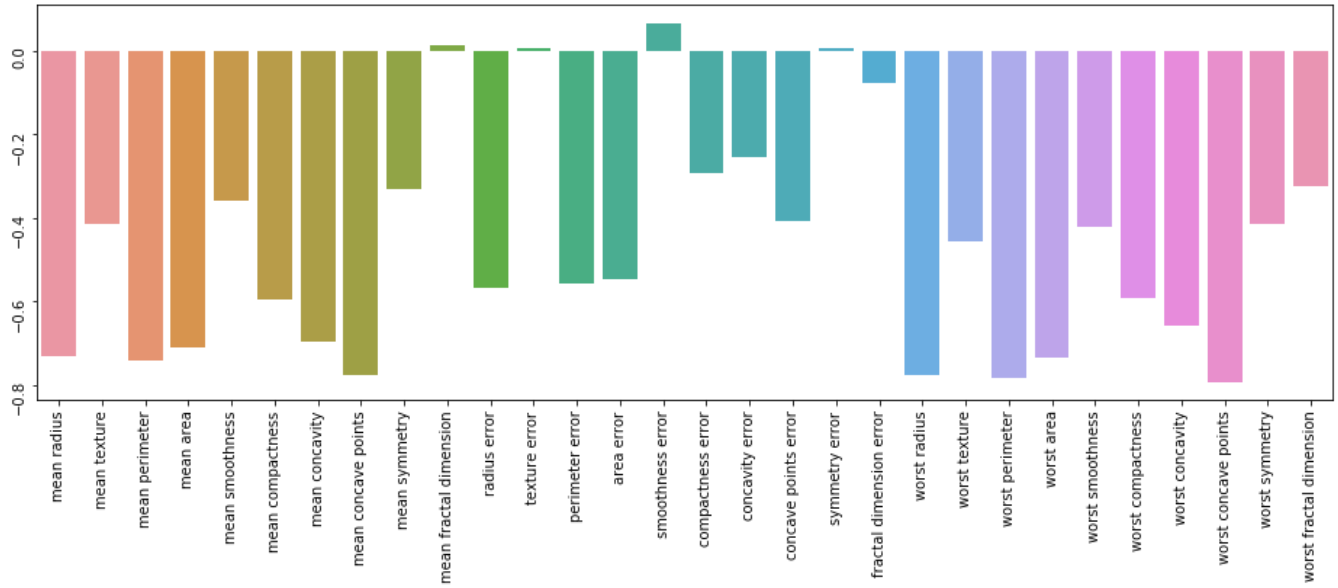



```
cancer_df2 = cancer_df.drop(['target'], axis = 1)
print("The shape of 'cancer_df2' is : ", cancer_df2.shape)
```

Output >>> The shape of 'cancer_df2' is : (569, 30)

```
plt.figure(figsize = (16,5))
ax = sns.barplot(cancer_df2.corrwith(cancer_df.target).index,
cancer_df2.corrwith(cancer_df.target))
ax.tick_params(labelrotation = 90)
```

Output >>>



2.3 Data Prepocessing Technique

Data preprocessing is a fundamental step in preparing data for machine learning algorithms. In the context of breast cancer prediction, where accurate and reliable data is crucial, proper preprocessing ensures that the data is clean, consistent, and suitable for training classification models.

```
1
2    X = cancer_df.drop(['target'], axis = 1)
3    X.head(6)
```

Output >>>

	mean radius	mean texture	mean perimeter	mean area	mean smoothness	mean compactness	mean concavity	mean concave points	mean symmetry	mean fractal dimension	...	worst radius	worst texture	worst perimeter	worst area	worst smoothness	worst compactness	worst concavity	worst concave points	worst symmetry	worst fractal dimension
0	17.99	10.38	122.80	1001.0	0.11840	0.27760	0.3001	0.14710	0.2419	0.07871	...	25.38	17.33	184.60	2019.0	0.1622	0.6656	0.7119	0.2654	0.4601	0.11890
1	20.57	17.77	132.90	1326.0	0.08474	0.07864	0.0869	0.07017	0.1812	0.05667	...	24.99	23.41	158.80	1956.0	0.1238	0.1866	0.2416	0.1860	0.2750	0.08902
2	19.69	21.25	130.00	1203.0	0.10960	0.15990	0.1974	0.12790	0.2069	0.05999	...	23.57	25.53	152.50	1709.0	0.1444	0.4245	0.4504	0.2430	0.3613	0.08758
3	11.42	20.38	77.58	386.1	0.14250	0.28390	0.2414	0.10520	0.2597	0.09744	...	14.91	26.50	98.87	567.7	0.2098	0.8663	0.6869	0.2575	0.6638	0.17300
4	20.29	14.34	135.10	1297.0	0.10030	0.13280	0.1980	0.10430	0.1809	0.05883	...	22.54	16.67	152.20	1575.0	0.1374	0.2050	0.4000	0.1625	0.2364	0.07678
5	12.45	15.70	82.57	477.1	0.12780	0.17000	0.1578	0.08089	0.2087	0.07613	...	15.47	23.75	103.40	741.6	0.1791	0.5249	0.5355	0.1741	0.3985	0.12440

6 rows × 30 columns

```
1
2    y = cancer_df['target']
3    y.head(6)
```

Output >>>

```
1    0    0.0
2    1    0.0
     2    0.0
```

```
3    3    0.0
4    4    0.0
5    5    0.0
6    Name: target, dtype: float64
7
```

```
1    from sklearn.model_selection import train_test_split
2    X_train, X_test, y_train, y_test =
3     train_test_split(X, y, test_size = 0.2, random_state= 5)
```

2.4 Feature Extraction Technique

Feature extraction is a crucial step in preparing data for machine learning models. In the context of breast cancer prediction, relevant and informative features can significantly impact the accuracy of the classification model.Converting different units and magnitude data in one unit.

```
1
2    from sklearn.preprocessing import StandardScaler
3    sc = StandardScaler()
4    X_train_sc = sc.fit_transform(X_train)
5    X_test_sc = sc.transform(X_test)
```

2.5 Classification Algorithms

We will used these algorithms to train and predict on your breast cancer dataset. The accuracy results suggest how well these models are performing on our test data.

```
1    from sklearn.metrics import confusion_matrix, classification_report, accuracy_score
```

Support Vector Classifier

```
1    from sklearn.svm import SVC
2    svc_classifier = SVC()
3    svc_classifier.fit(X_train, y_train)
4    y_pred_scv = svc_classifier.predict(X_test)
5    accuracy = accuracy_score(y_test, y_pred_scv).round(4)
6
     print("The accuracy of the SVM is:", accuracy)
```

Output >>> The accuracy of the SVM is: 0.9386

K – Nearest Neighbor Classifier

```
from sklearn.neighbors import KNeighborsClassifier
knn_classifier = KNeighborsClassifier(n_neighbors = 5, metric = 'minkowski', p = 2)
knn_classifier.fit(X_train, y_train)
y_pred_knn = knn_classifier.predict(X_test)
accuracy = accuracy_score(y_test, y_pred_knn).round(4)

print("The accuracy of the knn is:", accuracy)
```

Output >>> The accuracy of the knn is: 0.9386

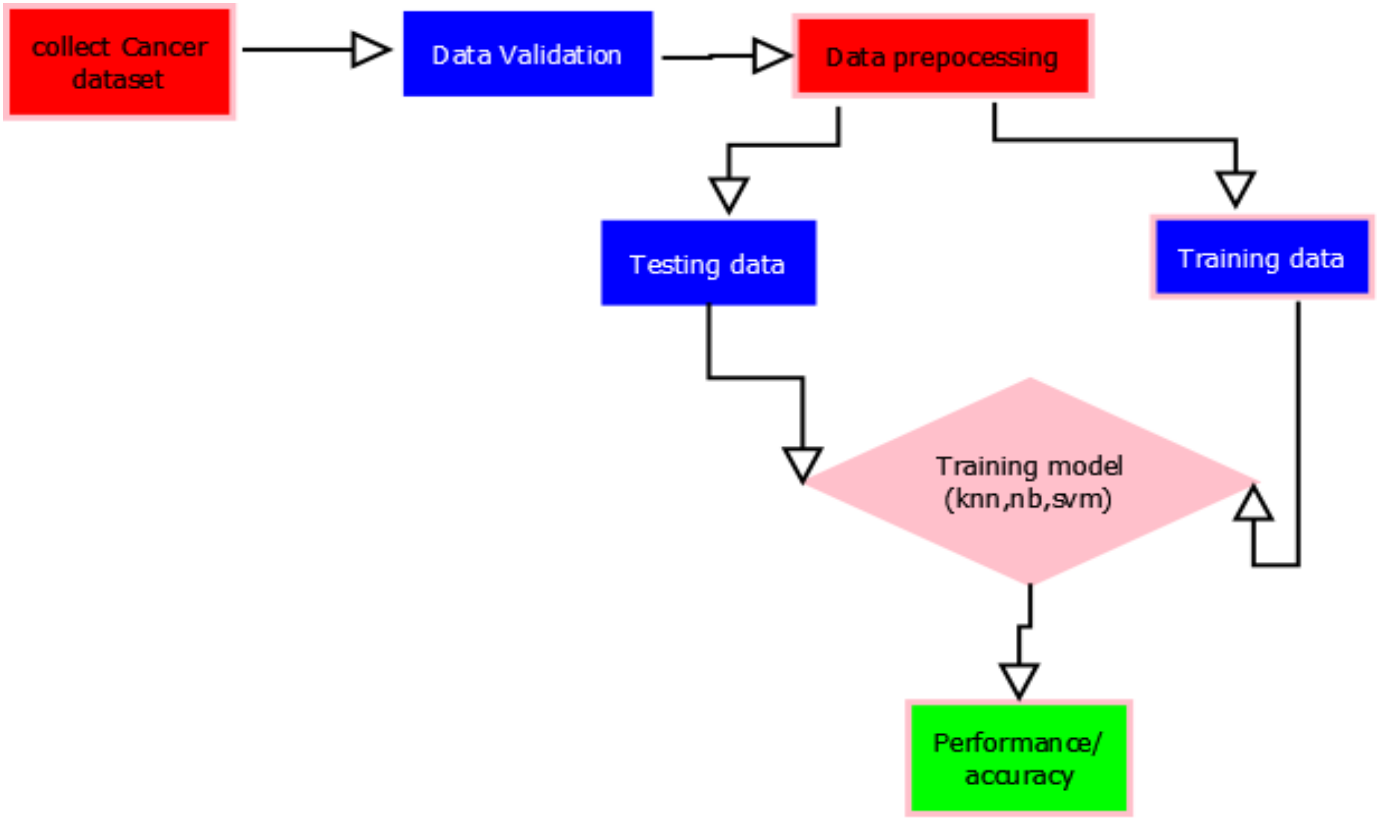
Naive Bayes Classifier

```
from sklearn.naive_bayes import GaussianNB
nb_classifier = GaussianNB()
nb_classifier.fit(X_train, y_train)
y_pred_nb = nb_classifier.predict(X_test)
accuracy = accuracy_score(y_test, y_pred_nb).round(4)

print("The accuracy of the nb is:", accuracy)
```

Output >>> The accuracy of the nb is: 0.9474

2.6 Block Diagram of Proposed Model



2.7 Data Analysis Techniques

One of the key components of our research on breast cancer was analyzing the data we collected. We used a variety of techniques to gain insights and draw conclusions from the data. One such technique was exploratory data analysis, which allowed us to identify patterns and trends in the data that we could use to inform our models. Another important technique we used was feature selection. This involved identifying the most relevant features in the data set that were most predictive of the outcome variable, in this case, whether a patient had breast cancer or not. By selecting only the most relevant features, we were able to improve the accuracy of our models and reduce the risk of overfitting.

2.8 Experimental Setup

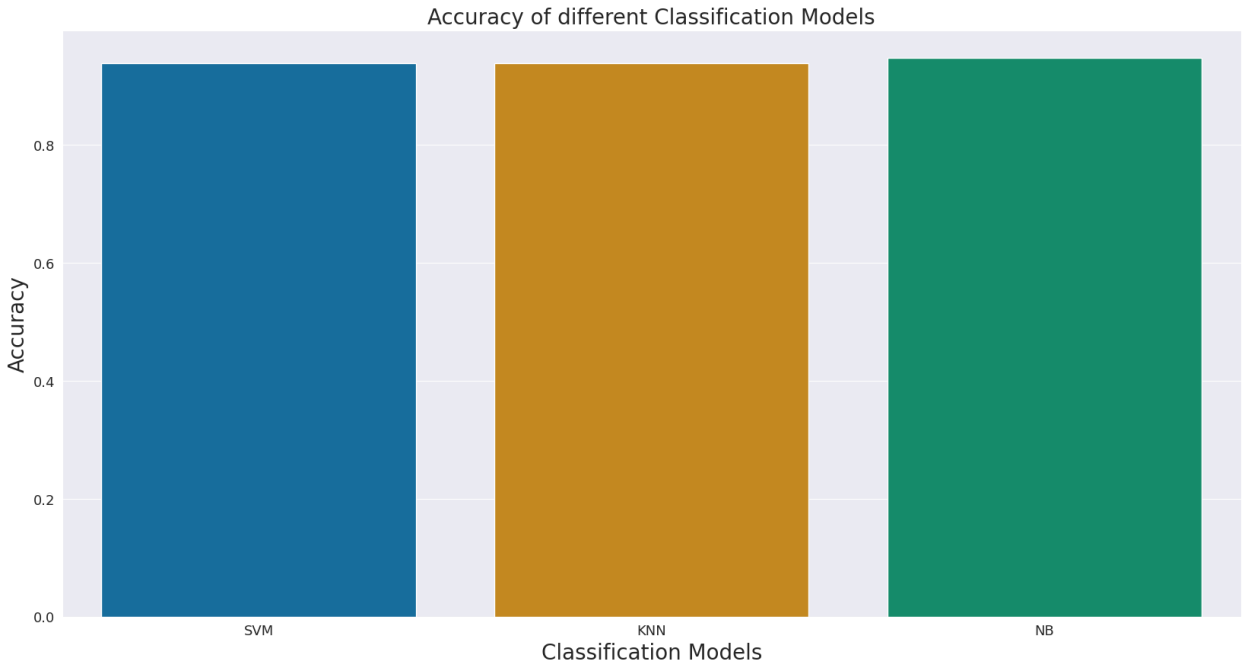
In order to conduct our experiments, we used a dataset of breast cancer patients that had been previously diagnosed. The dataset included various attributes such as age, tumor size, and lymph node involvement. We then preprocessed the data by removing any missing values and normalizing the features. Next, we split the dataset into training and testing sets. We then applied three different classification models - KNN, SVM, and NB - on the training set to predict whether a patient has breast cancer or not. We evaluated the performance of each model using various metrics such as accuracy, precision, and recall.

3. Results and Discussion

3.1 Results Analysis by comparison existing solution

In this report, we did an analysis on a dataset known as the 'load_breast_cancer'. Here, we developed 3 different of classifier model which are Support Vector Machine, k Nearest Neighbours, and Gaussian Naive Bayes. We can see that the accuracy model is K Nearest Neighbours (KNN) which accuracy is 0.9386. The Support Vector Machine (SVM) acuracy is 0.9386, and Gaussian Naive Bayes (NB) accuracy is 0.9474. So, the highest accuracy rate is 0.9474 which is Gaussian Naive Bayes (NB) classifier model. As a result, we can say that the Gaussian Naive Bayes classifier is the best use for this dataset model. The Gaussian Naive Bayes classifier model's accuracy is above 94% because of the dataset. It might perform better if we can train these model on a larger dataset.So in our opinion, depending on this dataset, the Gaussian Naive Bayes classifier is best for predicting the breast cancer . Although, for a larger dataset other model may perform better.

3.2 Results validation by Graphical Representation



4. Conclusion and Future Recommendations

4.1 Significant of outcomes

The outcomes of our study provide valuable insights into the effectiveness of different classification models for breast cancer detection. Our results show that the KNN model had the highest accuracy rate, followed closely by the SVM model. The NB model, while still effective, had a lower accuracy rate compared to the other two models. Furthermore, our data analysis techniques allowed us to identify key features and patterns in the data that were indicative of breast cancer. This information can be used to improve the accuracy of future classification models and ultimately aid in early detection and treatment of breast cancer.

4.2 Recommendation for Future Work

One of the key recommendations for future work is to explore the potential of incorporating other data sources into our analysis. While our current approach has yielded promising results, there may be additional variables that could further improve the accuracy and precision of our classification model.Another area for future research is to investigate the effectiveness of different feature selection techniques. We utilized a standard approach in our analysis, but there may be alternative methods that could yield better performance.