VitalCare

HealthCare System With ML Powered Prediction model

Submitted by

SAHIL KOSHRIYA

Roll No: 22223110
MCA- V Semester
(Session: 2022-2025)

Submitted to

DR. PRIYANKA TRIPATHI

(Head)

Department of Computer Applications
National Institute of Technology Raipur

ASSIST.PROF. SHARBANI PURKASHTHA

(Supervisor)

Department of Computer Applications
National Institute of Technology Raipur

Department of Computer Applications
National Institute of Technology, Raipur

Department of Computer Applications (संगणक उपयोजन विभाग)
National Institute of Technology Raipur (राष्ट्रीय प्रौद्योगिकी संस्थान रायपुर)
(An Institute of National Importance, Govt. of India)
G.E. Road, Raipur, Chhattisgarh, India, Pin – 492010
www.nitrr.ac.in

DECLARATION

I, **Sahil Koshriya, 22223110**, hereby declare that the work done in the project entitled **Vital Care** is done on my own.I

confirm that:

- The work contained in this report is original and has been done by me under the guidance of ASSIST.PROF. SHARBANI PURKASTHA, Department of Computer Applications, National Institute of Technology Raipur.
- The work has not been submitted to any other institute for any other degree or diploma;
- I have followed the guidelines provided by the institute in preparing the project report;
- I have conformed to ethical norms and guidelines while writing the project report.
- Whenever I have used materials such as data, models, figures, and text from other sources, I have given them due credit by citing them in the report and providing their details in the references.

Place: Raipur

Date:

Student name and signature

Roll No:22223110

MCA-V Semester

____

Department of Computer Applications (संगणक उपयोजन विभाग)
National Institute of Technology Raipur (राष्ट्रीय प्रौद्योगिकी संस्थान रायपुर)
(An Institute of National Importance, Govt. of India)
G.E. Road, Raipur, Chhattisgarh, India, Pin – 492010
www.nitrr.ac.in

Certificate from Organization

CERTIFICATE FROM THE SUPERVISOR

This is to certify that the project entitled____**Vital Care**______has been carried out by **Sahil Koshriya, 22223110.**, MCA 5th Semester, under my guidance.

The matter embodied in this project has not been submitted for the award of any other degree or diploma to the best of my knowledge.

Place: Raipur

Date:

____

(Supervisor signature and seal)

Acknowledgement

I would like to express my sincere thanks to **Prof. Sharbani Purkashtha Mam, Assistant Professor**, National Institute of Technology Raipur, for his valuable guidance and support in completing my project. I would also like to express my gratitude towards our Professor & Head of Department **Dr Priyanka Tripathi (HOD)**for giving me this great opportunity to do a projecton Health Care Platform. Without their support and suggestions, this project would not have been completed.

SAHIL KOSHRIYA

22223110

Report # 2

Female Awareness Breast Cancer Prediction Using Machine Learning

Description:

Breast cancer remains a significant health challenge globally, being one of the leading causes of cancer-related mortality among women. Early detection is critical for effective treatment, and machine learning (ML) has emerged as a powerful tool in medical diagnostics to help address this challenge. Machine learning models have the potential to analyze vast amounts of data, recognize intricate patterns, and provide accurate predictions that can assist healthcare professionals in making more informed decisions.

In this work, we introduce a machine learning-based approach to breast cancer prediction, leveraging patient data such as imaging features, clinical history, and demographic information. By utilizing advanced algorithms like decision trees, support vector machines, and neural networks, our goal is to develop a robust and reliable model capable of distinguishing between benign and malignant cases. The use of ML not only aids in improving diagnostic accuracy but also helps reduce false positives and false negatives, which are crucial for ensuring effective patient management.

The focus of this study is to demonstrate the applicability of machine learning techniques in building predictive models that can complement traditional diagnostic methods. By harnessing the power of data-driven insights, we aim to support healthcare providers in making timely and accurate decisions, ultimately contributing to better patient outcomes and a higher survival rate for breast cancer patients.

Models Algo we Can Use

1. **Decision Trees**: A tree-like model that splits the dataset based on features to make predictions. It is easy to interpret and provides a clear visualization of decision-making.
2. **Support Vector Machines (SVM)**: A powerful classification algorithm that works well for high-dimensional datasets. It aims to find the optimal hyperplane that separates different classes.
3. **Neural Networks**: Deep learning models that are capable of learning complex relationships within the data. Specifically, we use multilayer perceptrons (MLP) to distinguish between benign and malignant cases effectively.
4. **Random Forest**: An ensemble learning technique that uses multiple decision trees to improve accuracy and prevent overfitting.
5. **Logistic Regression**: A baseline model for binary classification tasks. It is simple yet effective for predicting the probability of malignancy.
6. **K-Nearest Neighbors (KNN)**: A non-parametric method that classifies a data point based on the majority class of its nearest neighbors. It is useful for comparison and baseline purposes.

MODEL 2: Breast Cancer Prediction Using Python

Importing libraries

```
# importing libraries
import numpy
import matplotlib.pyplot as plt
import pandas as pd
import seaborn as sns

# reading data from the file
df=pd.read_csv("data.csv")

df.head()

{"type":"dataframe","variable_name":"df"}


df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 569 entries, 0 to 568
Data columns (total 33 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
 32  Unnamed: 32              0 non-null      float64
dtypes: float64(31), int64(1), object(1)
memory usage: 146.8+ KB
```

```
# return the size of dataset
df.shape
```

```
(569, 33)
```

```
# remove the column
df=df.dropna(axis=1)
```

```
# shape of dataset after removing the null column
df.shape
```

```
(569, 32)
```

```
# describe the dataset
df.describe()
```

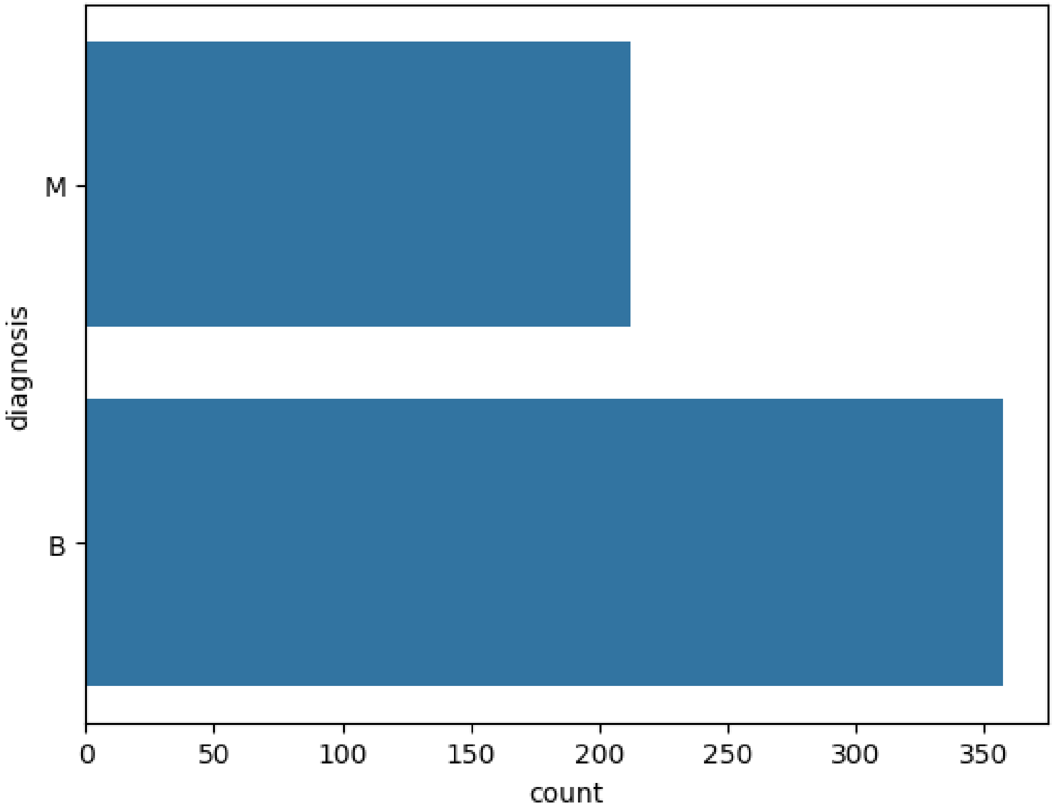
```
{"type":"dataframe"}
```

```
# Get the count of malignant<M> and Benign<B> cells
df['diagnosis'].value_counts()
```

```
diagnosis
B    357
M    212
Name: count, dtype: int64
```

```
sns.countplot(df['diagnosis'],label="count")
```

```
<Axes: xlabel='count', ylabel='diagnosis'>
```



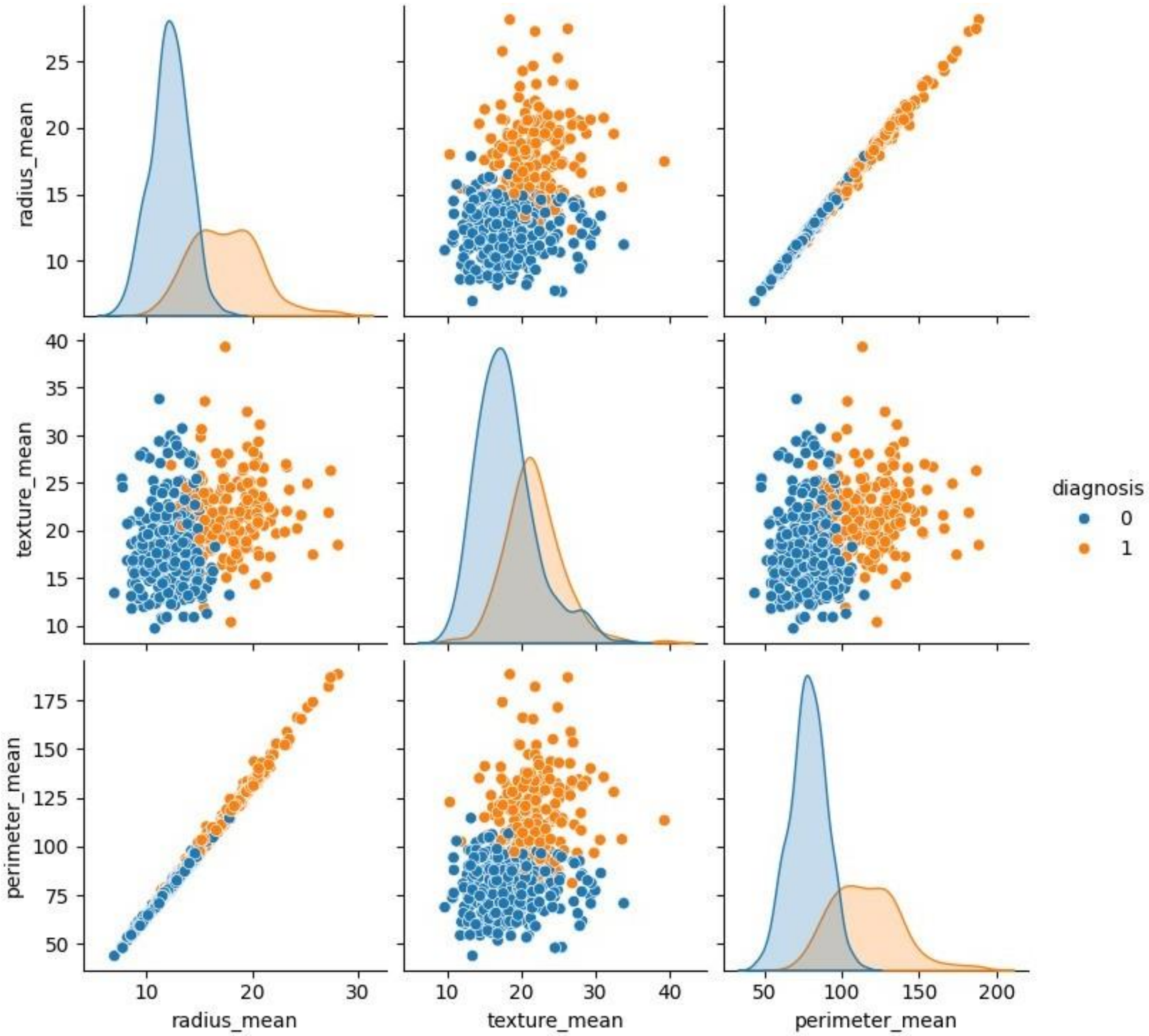
```
# label encoding(convert the value of M and B into 1 and 0)
from sklearn.preprocessing import LabelEncoder
labelencoder_Y = LabelEncoder()
df.iloc[:,1]=labelencoder_Y.fit_transform(df.iloc[:,1].values)


df.head()
```

{"type":"dataframe","variable_name":"df"}

```
sns.pairplot(df.iloc[:,1:5],hue="diagnosis")
```

```
<seaborn.axisgrid.PairGrid at 0x7af9d51ebac0>
```

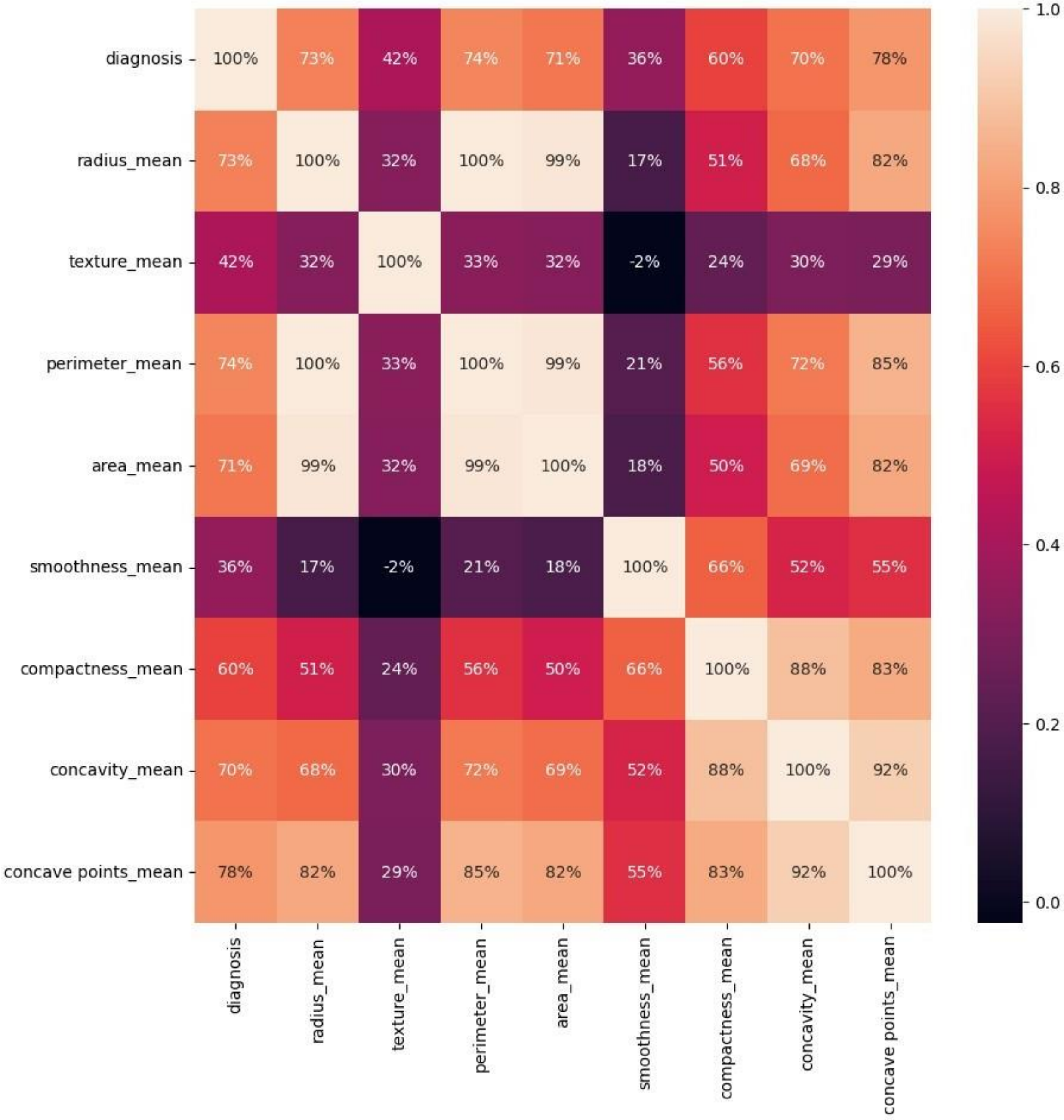


```
# get the correlation
df.iloc[:,1:32].corr()
```

```
{"type":"dataframe"}
```

```
# visualize the correlation
plt.figure(figsize=(10,10))
sns.heatmap(df.iloc[:,1:10].corr(),annot=True,fmt=".0%")
```

```
<Axes: >
```



	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean	concavity_mean	concave points_mean
diagnosis	100%	73%	42%	74%	71%	36%	60%	70%	78%
radius_mean	73%	100%	32%	100%	99%	17%	51%	68%	82%
texture_mean	42%	32%	100%	33%	32%	-2%	24%	30%	29%
perimeter_mean	74%	100%	33%	100%	99%	21%	56%	72%	85%
area_mean	71%	99%	32%	99%	100%	18%	50%	69%	82%
smoothness_mean	36%	17%	-2%	21%	18%	100%	66%	52%	55%
compactness_mean	60%	51%	24%	56%	50%	66%	100%	88%	83%
concavity_mean	70%	68%	30%	72%	69%	52%	88%	100%	92%
concave points_mean	78%	82%	29%	85%	82%	55%	83%	92%	100%

```
# split the dataset into dependent(X) and Independent(Y) datasets
X=df.iloc[:,2:31].values
Y=df.iloc[:,1].values

# spliting the data into trainning and test dateset
from sklearn.model_selection import train_test_split
X_train,X_test,Y_train,Y_test=train_test_split(X,Y,test_size=0.20,rand
om_state=0)

# feature scaling
from sklearn.preprocessing import StandardScaler
```

```
X_train=StandardScaler().fit_transform(X_train)
X_test=StandardScaler().fit_transform(X_test)

# models/ Algorithms

def models(X_train,Y_train):
        #logistic regression
        from sklearn.linear_model import LogisticRegression
        log=LogisticRegression(random_state=0)
        log.fit(X_train,Y_train)


        #Decision Tree
        from sklearn.tree import DecisionTreeClassifier

tree=DecisionTreeClassifier(random_state=0,criterion="entropy")
        tree.fit(X_train,Y_train)

        #Random Forest
        from sklearn.ensemble import RandomForestClassifier

forest=RandomForestClassifier(random_state=0,criterion="entropy",n_est
imators=10)
        forest.fit(X_train,Y_train)

        print('[0]logistic regression
accuracy:',log.score(X_train,Y_train))
        print('[1]Decision tree
accuracy:',tree.score(X_train,Y_train))
        print('[2]Random forest
accuracy:',forest.score(X_train,Y_train))

        return log,tree,forest

model=models(X_train,Y_train)
```

```
[0]logistic regression accuracy: 0.9472527472527472
[1]Decision tree accuracy: 1.0
[2]Random forest accuracy: 1.0
```

```
/usr/local/lib/python3.10/dist-packages/sklearn/linear_model/
_logistic.py:469: ConvergenceWarning: lbfgs failed to converge
(status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as
shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:

https://scikit-learn.org/stable/modules/linear_model.html#logistic-
```

```
regression
  n_iter_i = _check_optimize_result(
```

```
# testing the models/result

from sklearn.metrics import accuracy_score
from sklearn.metrics import classification_report

for i in range(len(model)):
    print("Model",i)
    print(classification_report(Y_test,model[i].predict(X_test)))
    print('Accuracy :
',accuracy_score(Y_test,model[i].predict(X_test)))
```

```
Model 0
              precision    recall  f1-score   support

           0       0.97      0.91      0.94        43
           1       0.95      0.99      0.97        71

    accuracy                           0.96       114
   macro avg       0.96      0.95      0.95       114
weighted avg       0.96      0.96      0.96       114

Accuracy :  0.956140350877193
Model 1
              precision    recall  f1-score   support

           0       0.97      0.91      0.94        43
           1       0.95      0.99      0.97        71

    accuracy                           0.96       114
   macro avg       0.96      0.95      0.95       114
weighted avg       0.96      0.96      0.96       114

Accuracy :  0.956140350877193
Model 2
              precision    recall  f1-score   support

           0       0.98      0.93      0.95        43
           1       0.96      0.99      0.97        71

    accuracy                           0.96       114
   macro avg       0.97      0.96      0.96       114
weighted avg       0.97      0.96      0.96       114

Accuracy :  0.9649122807017544
```

```
# prediction of random-forest
pred=model[2].predict(X_test)
print('Predicted values:')
print(pred)
print('Actual values:')
print(Y_test)
```

```
Predicted values:
[1 0 0 1 1 0 0 0 0 1 1 0 1 0 1 0 1 1 1 0 1 1 0 1 1 1 1 1 1 0 1 1 1 1 1
1 0
 1 0 1 1 0 1 1 1 1 1 1 1 1 0 0 1 1 1 1 1 0 0 1 1 0 0 1 1 1 0 0 1 1 0 0
1 0
 1 1 1 1 1 1 0 1 1 0 0 0 0 0 1 1 1 1 1 1 1 1 0 0 1 0 0 1 0 0 1 1 1 0 1
1 0
 1 1 0]
Actual values:
204    1
70     0
131    0
431    1
540    1
      ..
486    1
75     0
249    1
238    1
265    0
Length: 114, dtype: int64
```

```
from joblib import dump
dump(model[2],"Feamle_Awareness_Breast_Cancer_prediction.joblib")
```

```
['Feamle_Awareness_Breast_Cancer_prediction.joblib']
```