

## Breast Cancer Prediction Model



Shreya Bhattacharjee  
6861  
Machine Learning Project

MAHATMA EDUCATION SOCIETY'S  
PILLAI COLLEGE OF ARTS, COMMERCE & SCIENCE  
(Autonomous)  
NEW PANVEL

PROJECT REPORT ON  
**“Breast Cancer Risk Prediction Model”**

IN PARTIAL FULFILLMENT OF

MASTER OF DATA ANALYTICS

SEMESTER III– 2024-25

PROJECT GUIDE

Name: Omkar Sherkhane

SUBMITTED BY: Shreya Bhattacharjee

ROLL NO: 6861

Mahatma Education Society's  
**PILLAI COLLEGE OF ARTS, COMMERCE & SCIENCE**  
(Autonomous)  
Re-accredited "A" Grade by NAAC (3<sup>rd</sup> Cycle)



**Project Completion Certificate**  
**THIS IS TO CERTIFY THAT**

**Shreya Bhattacharjee**

of **M.Sc. Data Analytics Part - II** has completed the project titled **Breast Cancer Prediction Model** of subject **Machine Learning** under our guidance and supervision during the academic year 2024-25 in the department of Master of Data Analytics.

Project Guide    Course Coordinator    Head of the Department

## **Introduction:**

Breast cancer is one of the most prevalent and life-threatening diseases affecting women worldwide. Early detection and accurate diagnosis are crucial in improving the chances of successful treatment and survival. The Breast Cancer Prediction Model is a machine learning-based application developed to aid in the early detection of breast cancer by predicting the likelihood of a patient having malignant (cancerous) or benign (non-cancerous) tumors based on their medical features.

This project leverages advanced classification algorithms to analyze medical data and provide reliable predictions, which can assist healthcare professionals in making informed decisions. The model is trained on the Breast Cancer Wisconsin (Diagnostic) dataset, a widely-used dataset in the medical research community that contains detailed measurements of cell nuclei present in breast cancer biopsies.

The predictive model is deployed within an easy-to-use web application, built using the Flask framework. This web interface allows users to input specific medical features related to breast cancer, and it returns predictions along with informative messages and visual aids to help interpret the results.

The Breast Cancer Prediction Model demonstrates the potential of machine learning in healthcare, showcasing how technology can be harnessed to enhance diagnostic accuracy and support early intervention strategies. By providing a user-friendly platform for breast cancer prediction, this project aims to contribute to the ongoing efforts to improve breast cancer outcomes and save lives through early detection.

# Technologies Used for the Breast Cancer Prediction Model

## 1. Python

- **Description:** A high-level programming language known for its readability and versatility.
- **Role:** Used for data processing, model training, and application development. Python's extensive libraries facilitate machine learning and web development.

## 2. Scikit-learn

- **Description:** A powerful machine learning library for Python.
- **Role:** Provides tools for building and training the predictive model. It includes various algorithms for classification, such as support vector machines, decision trees, and ensemble methods.

## 3. Flask

- **Description:** A lightweight web framework for Python.
- **Role:** Used to build the web application that allows users to interact with the predictive model. Flask handles web requests, routing, and rendering of HTML templates.

## 4. Pickle

- **Description:** A Python module for serializing and deserializing Python objects.
- **Role:** Used to save the trained machine learning model to a file (`model.pkl`) and load it into the application for making predictions.

## 5. HTML/CSS

- **Description:** Standard technologies for creating and styling web pages.
- **Role:** HTML is used for structuring the content of the web application, while CSS is used for styling and visual design, including layout and color schemes.

## 6. Bootstrap

- **Description:** A front-end framework for developing responsive and mobile-first websites.
- **Role:** Provides pre-designed components and styles to create a responsive and visually appealing user interface. Bootstrap simplifies the design process and ensures compatibility across various devices.

## 7. Jinja2

Shreya Bhattacharjee  
6861  
Machine Learning Project

- **Description:** A templating engine for Python, used with Flask.
- **Role:** Allows dynamic generation of HTML pages by embedding Python code within HTML templates. It helps in rendering data and results from the Flask application into the web pages.

## 8. NumPy

- **Description:** A fundamental package for scientific computing with Python.
- **Role:** Provides support for large multi-dimensional arrays and matrices. Used to handle and manipulate the input features for prediction.

## 9. Pandas

- **Description:** A data manipulation and analysis library for Python.
- **Role:** Facilitates data cleaning, transformation, and analysis. It is particularly useful for preprocessing the dataset before training the model.

## 10. Matplotlib & Seaborn

- **Description:** Libraries for data visualization in Python.
- **Role:** Used to create plots and visualizations that help in analyzing the data and understanding model performance.

## CODE AND OUTPUT

```
import numpy as np
import pandas as pd
```

```
breast=pd.read_csv('/content/breast_cancer.csv')
breast.head()
```

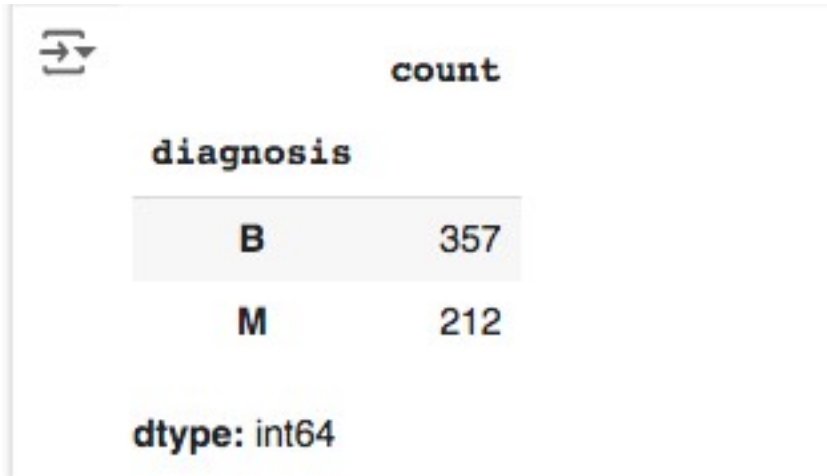
	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean	concavity_mean	concave points_mean	...	texture_v
0	842302	M	17.99	10.38	122.80	1001.0	0.11840	0.27760	0.3001	0.14710	...	
1	842517	M	20.57	17.77	132.90	1326.0	0.08474	0.07864	0.0869	0.07017	...	
2	84300903	M	19.69	21.25	130.00	1203.0	0.10960	0.15990	0.1974	0.12790	...	
3	84348301	M	11.42	20.38	77.58	386.1	0.14250	0.28390	0.2414	0.10520	...	
4	84358402	M	20.29	14.34	135.10	1297.0	0.10030	0.13280	0.1980	0.10430	...	

5 rows x 33 columns

```
breast.shape
```

```
(569, 33)
```

```
breast['diagnosis'].value_counts()
```



The image shows a Jupyter Notebook output for the command `breast['diagnosis'].value_counts()`. It features a table with two columns: **diagnosis** and **count**. The first row shows 'B' with a count of 357, and the second row shows 'M' with a count of 212. Below the table, it indicates the data type is `dtype: int64`. A Jupyter icon is visible in the top left corner of the output area.

diagnosis	count
B	357
M	212

dtype: int64

```
breast.info()
```



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

```
breast.isnull().sum()
```



	0
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

```
# Convert 'diagnosis' column to numerical
representation
breast['diagnosis'] = breast['diagnosis'].map({'M': 1,
'B': 0})

# Now calculate correlations
breast.corr()
```

	id	diagnosis	radius_mean	texture_mean
id	1.000000	0.039769	0.074626	0.099770
diagnosis	0.039769	1.000000	0.730029	0.415185
radius_mean	0.074626	0.730029	1.000000	0.323782
texture_mean	0.099770	0.415185	0.323782	1.000000
perimeter_mean	0.073159	0.742636	0.997855	0.329533
area_mean	0.096893	0.708984	0.987357	0.321086
smoothness_mean	-0.012968	0.358560	0.170581	-0.023389
compactness_mean	0.000096	0.596534	0.506124	0.236702
concavity_mean	0.050080	0.696360	0.676764	0.302418

```
breast.drop('Unnamed: 32', axis=1, inplace=True)
```

```
breast.describe()
```

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean
count	5.690000e+02	569.000000	569.000000	569.000000	569.000000	569.000000
mean	3.037183e+07	0.372583	14.127292	19.289649	91.969033	654.889104
std	1.250206e+08	0.483918	3.524049	4.301036	24.298981	351.914129
min	8.670000e+03	0.000000	6.981000	9.710000	43.790000	143.500000
25%	8.692180e+05	0.000000	11.700000	16.170000	75.170000	420.300000
50%	9.060240e+05	0.000000	13.370000	18.840000	86.240000	551.100000
75%	8.813129e+06	1.000000	15.780000	21.800000	104.100000	782.700000
max	9.113205e+08	1.000000	28.110000	39.280000	188.500000	2501.000000

8 rows x 32 columns

```
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
```

```
breast['diagnosis'] =  
le.fit_transform(breast['diagnosis'])
```

## Splitting Data into Training and Testing Sets

```
from sklearn.model_selection import train_test_split  
# Split the data into training and testing sets  
X = breast.drop('diagnosis', axis=1)  
y = breast['diagnosis']  
X_train, X_test, y_train, y_test = train_test_split(X,  
y, test_size=0.2, random_state=42)
```

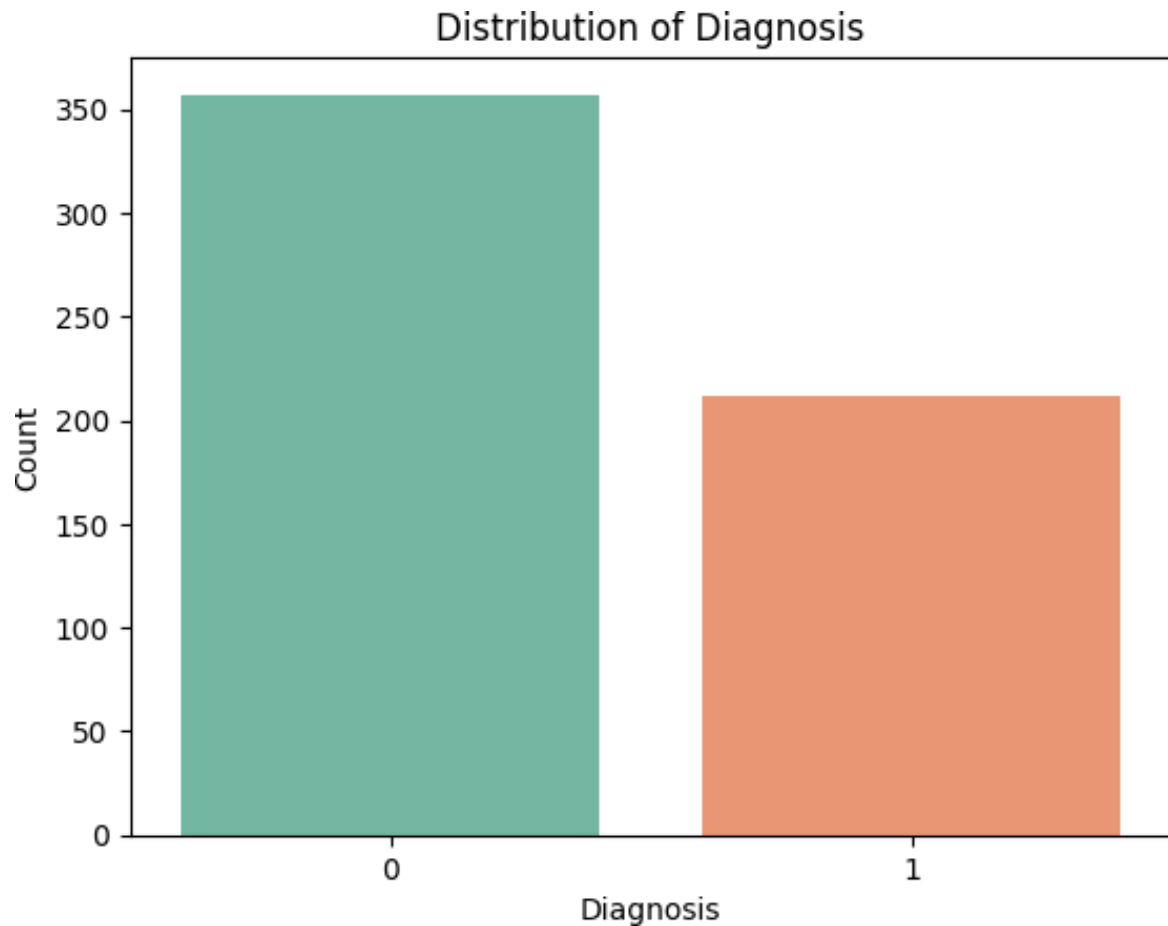
## Feature Scaling

```
from sklearn.preprocessing import StandardScaler  
# Scale the features  
scaler = StandardScaler()  
X_train = scaler.fit_transform(X_train)  
X_test = scaler.transform(X_test)
```

## Exploratory Data Analysis (EDA)

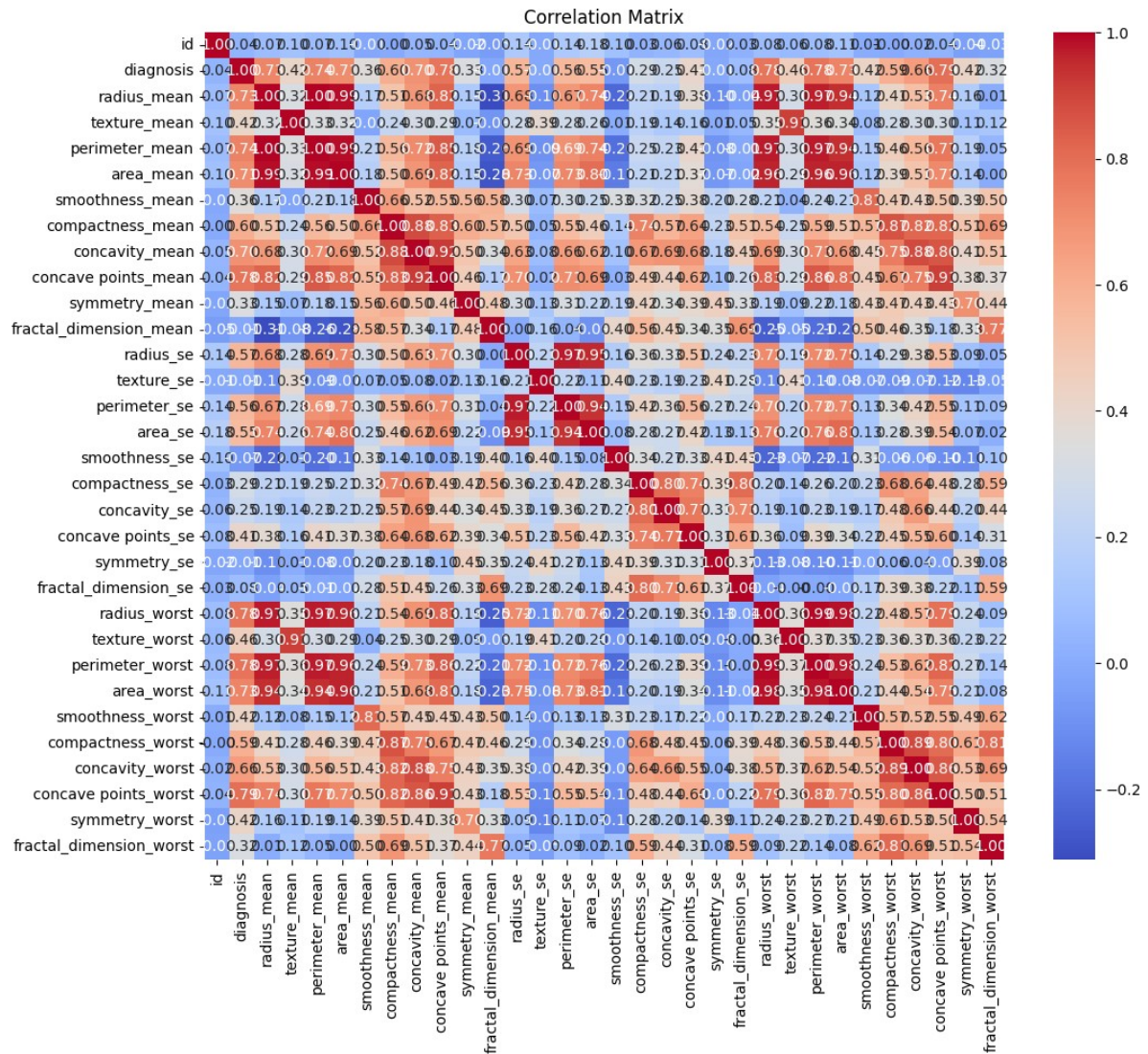
```
import seaborn as sns  
import matplotlib.pyplot as plt  
  
# Assuming 'breast' is a DataFrame and 'diagnosis' is  
# the target variable  
# Set a specific palette to differentiate the bars  
sns.countplot(x=breast['diagnosis'], palette='Set2')  
plt.title('Distribution of Diagnosis')
```

```
plt.xlabel('Diagnosis')  
plt.ylabel('Count')  
plt.show()
```



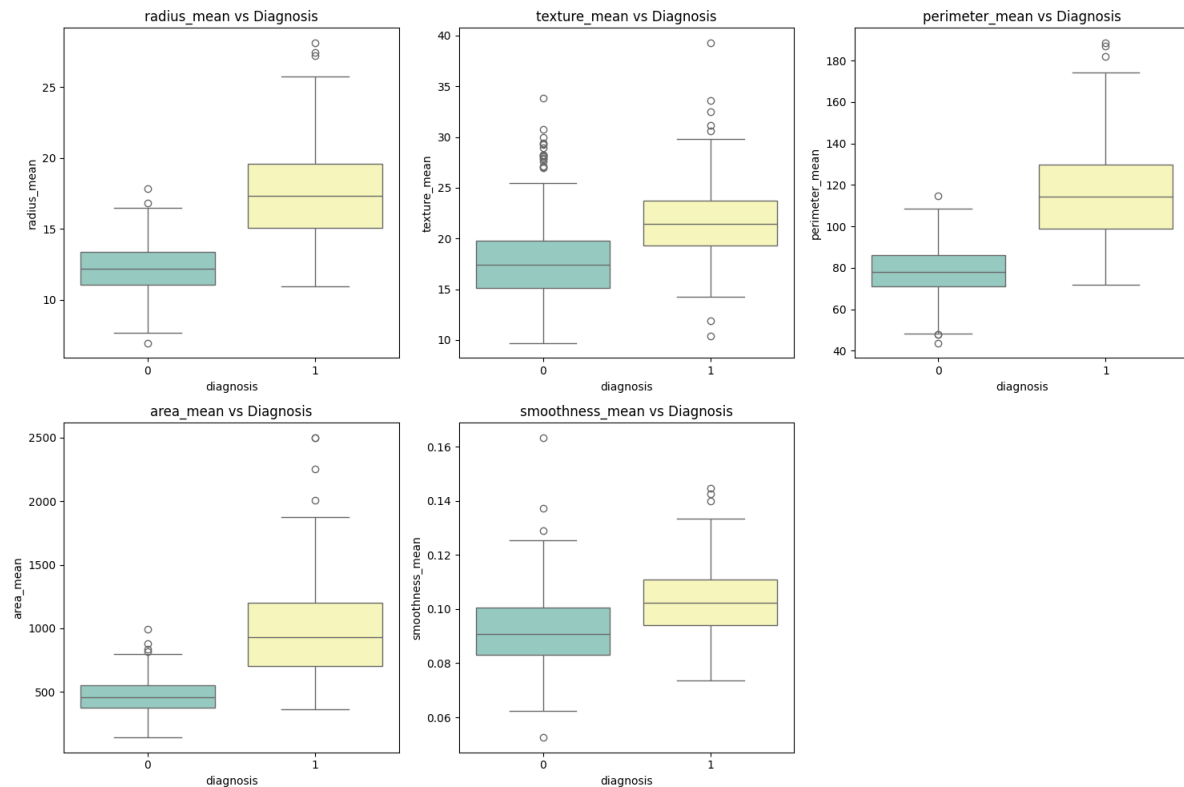
```
# Correlation Heatmap  
plt.figure(figsize=(12, 10))  
sns.heatmap(breast.corr(), annot=True, cmap='coolwarm',  
fmt=".2f")  
plt.title('Correlation Matrix')  
plt.show()
```



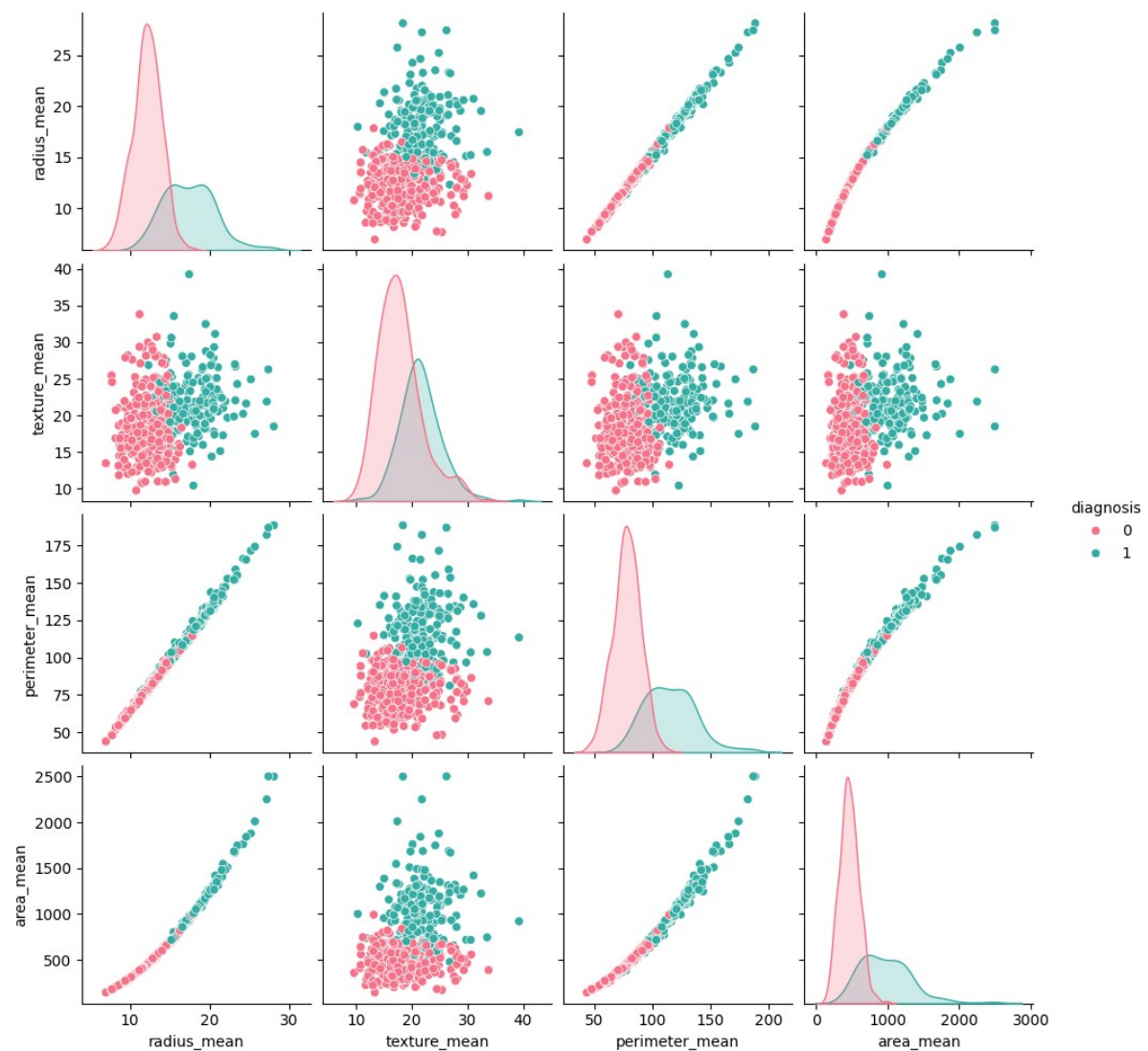


```
# Box Plots for Selected Features
features_to_plot = ['radius_mean', 'texture_mean',
                    'perimeter_mean', 'area_mean', 'smoothness_mean']
plt.figure(figsize=(15, 10))
for i, feature in enumerate(features_to_plot, 1):
    plt.subplot(2, 3, i)
```

```
sns.boxplot(x='diagnosis', y=feature, data=breast,  
palette='Set3')  
plt.title(f'{feature} vs Diagnosis')  
plt.tight_layout()  
plt.show()
```



```
# Pairplot for Selected Features  
selected_features = ['radius_mean', 'texture_mean',  
                    'perimeter_mean', 'area_mean']  
sns.pairplot(breast[selected_features + ['diagnosis']],  
hue='diagnosis', palette='husl')  
plt.show()
```



```
# Visualize the correlation matrix  
breast.corr()
```



	id	diagnosis	radius_mean	texture_mean
id	1.000000	0.039769	0.074626	0.099770
diagnosis	0.039769	1.000000	0.730029	0.415185
radius_mean	0.074626	0.730029	1.000000	0.323782
texture_mean	0.099770	0.415185	0.323782	1.000000
perimeter_mean	0.073159	0.742636	0.997855	0.329533
area_mean	0.096893	0.708984	0.987357	0.321086
smoothness_mean	-0.012968	0.358560	0.170581	-0.023389
compactness_mean	0.000096	0.596534	0.506124	0.236702

## Training Model

```
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score

# Train the logistic regression model
lr = LogisticRegression()
lr.fit(X_train, y_train)

# Evaluate the performance on the testing set
y_pred_lr = lr.predict(X_test)
accuracy_lr = accuracy_score(y_test, y_pred_lr)
print('Accuracy:', accuracy_lr)
```

Accuracy: 0.9736842105263158

```
input_text = (-0.23717126, -0.64487029, -0.11382239, -
0.57427777, -0.60294971,
```

```
1.0897546, 0.91543814, 0.41448279, 0.09311633,  
1.78465117,  
2.11520208, 0.28454765, -0.31910982, 0.2980991,  
0.01968238,  
-0.47096352, 0.45757106, 0.28733283, -0.23125455,  
0.26417944,  
0.66325388, 0.12170193, 0.42656325, 0.36885508,  
0.02065602,  
1.39513782, 2.0973271, 2.01276347, 0.61938913,  
2.9421769,  
3.15970842)  
np_df = np.asarray(input_text)  
# Use 'lr' instead of 'lg' to make the prediction  
prediciont = lr.predict(np_df.reshape(1,-1))  
if prediciont[0] == 1:  
    print("Cancrous")  
else:  
    print("Not Cancrous")
```

Cancrous

### Píediction System

```
import numpy as np  
  
input_text = (-0.23717126, -0.64487029, -0.11382239, -  
0.57427777, -0.60294971,  
1.0897546 , 0.91543814, 0.41448279,  
0.09311633, 1.78465117,  
2.11520208, 0.28454765, -0.31910982,  
0.2980991 , 0.01968238,  
-0.47096352, 0.45757106, 0.28733283, -  
0.23125455, 0.26417944,  
0.66325388, 0.12170193, 0.42656325,  
0.36885508, 0.02065602,
```

```
1.39513782, 2.0973271 , 2.01276347,  
0.61938913, 2.9421769 ,  
3.15970842)  
np_df = np.asarray(input_text)  
pred = lr.predict(np_df.reshape(1,-1))  
if pred[0] == 1:  
    print("Cancrous")  
else:  
    print("Not Cancrous")
```

Saving a Machine Learning Model with Pickle

```
import pickle  
  
pickle.dump(lr, open('model.pkl','wb')) # Now pickle  
the 'lg' object
```

## FLASK CODE

### App.py

```
from flask import Flask, render_template, request
import numpy as np
import pandas as pd
import pickle

# loading model
model = pickle.load(open('model.pkl', 'rb'))

# flask app
app = Flask(__name__)

@app.route('/')
def home():
    return render_template('index.html')

@app.route('/predict', methods=['POST'])
def predict():
    features = request.form['feature']
    features = features.split(',')
    np_features = np.asarray(features, dtype=np.float32)

    # prediction
    pred = model.predict(np_features.reshape(1, -1))
    message = ['Cancrouse' if pred[0] == 1 else 'Not Cancrouse']
    # print(message[0])
    return render_template('index.html', message=message)

if __name__ == '__main__':
    app.run(debug=True)
```

## INDEX.HTML

```
<!doctype html>
<html lang="en">
  <head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <title>CA 2 FOR MACHINE LEARNING 6861</title>
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-
alpha3/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-
KK94CHFLLe+nY2dmCWGMq91rCGa5gtU4mk92HdvYe+M/SXH301p5ILy+dN9+nJOZ"
crossorigin="anonymous">
  </head>
  <body style="background:black; color:white;">

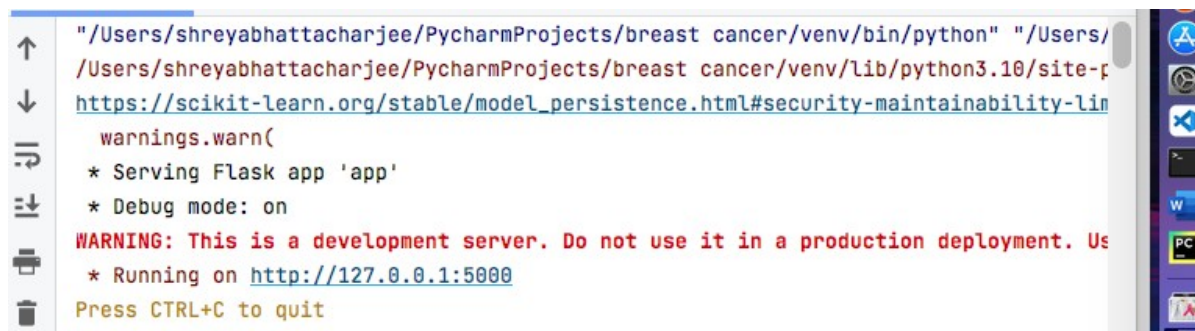
    <div class="container my-3 mt-3">
      <h1 style="text-align:center">Breast Cancer Prediction Model
(6861)</h1>
      
      <form action="/predict" method="POST">
        <div class="mb-3">
          <label for="text" class="form-label">Input Breast Cancer
Features</label>
          <input type="text" class="form-control" id="number"
name="feature" aria-describedby="emailHelp">
        </div>
        <button type="submit" class="btn btn-primary">Predict</button>
      </form>

      {% for i in message %}
      <p>{{i}}</p>
      {% if i=='Not Cancrouse' %}
        <div class="card my-3 mt-3" style="width: 18rem;">
          
          <div class="card-body">
            <h5 class="card-title">Not Cancrouse</h5>
            <p class="card-text">Don't Worry You don't have Breast
Cancer Disease, Enjoy your life.</p>
          </div>
        </div>
      {% else %}
        <div class="card my-3 mt-3" style="width: 18rem;">
          
          <div class="card-body">
            <h5 class="card-title">Cancrouse</h5>
            <p class="card-text">Alert You have a Breast Cancer
Disease, check yourself by doctor.</p>
          </div>
        </div>
      {% endif %}
    {% endfor %}

  </div>
```

```
<script
src="https://cdn.jsdelivr.net/npm/@popperjs/core@2.11.7/dist/umd/popper.min
.js" integrity="sha384-
zYPOMqeu1DAVkhILqWBUTcbYfZ8osulNd6Z89ify25QV9guujx43ITvfi12/QExE"
crossorigin="anonymous"></script>
<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-
alpha3/dist/js/bootstrap.min.js" integrity="sha384-
Y4oOpwW3duJdCWv5ly8SCFYWqFDsfob/3GkgExXKV4idmbt98QcxXYs9UoXAB7BZ"
crossorigin="anonymous"></script>
<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-
alpha3/dist/js/bootstrap.bundle.min.js" integrity="sha384-
ENjdO4Dr2bkBIFxQpeoTz1HIcje39Wm4jDKdf19U8gI4ddQ3GYNS7NTKfAdVQSZe"
crossorigin="anonymous"></script>
</body>
</html>
```

## Output

A screenshot of a terminal window with a light blue header bar. The terminal shows the output of a Flask application running in debug mode. The text is as follows:

```
"/Users/shreyabhattacharjee/PycharmProjects/breast cancer/venv/bin/python" "/Users/
/Users/shreyabhattacharjee/PycharmProjects/breast cancer/venv/lib/python3.10/site-p
https://scikit-learn.org/stable/model_persistence.html#security-maintainability-lin
warnings.warn(
* Serving Flask app 'app'
* Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Us
* Running on http://127.0.0.1:5000
Press CTRL+C to quit
```

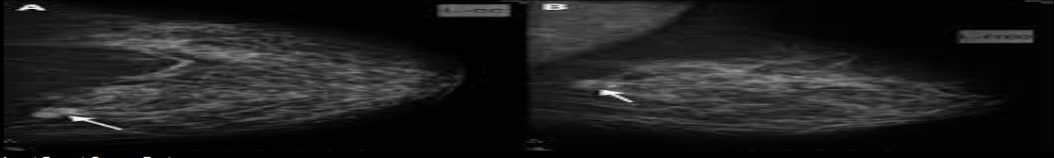
The terminal window has a vertical toolbar on the left with icons for back, forward, search, and other navigation functions. On the right side, a portion of the Windows taskbar is visible, showing icons for the Start menu, File Explorer, and other applications.

Shreya Bhattacharjee  
6861  
Machine Learning Project

127.0.0.1:5000/predict

PROJECT online f... Home - Canva Responsive Our S... Overview | Wix.com Hdmovie2.com W... Breast\_Cancer\_Us... ABC | Academic B... All Bookmarks

## Breast Cancer Prediction Model (6861)

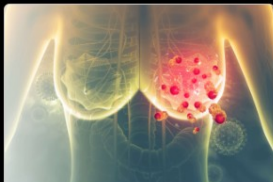


Input Breast Cancer Features

-0.23717126, -0.64487029, -0.11382239, -0.57427777, -0.60294971, 1.0897546, 0.91543814, 0.41448279, 0.09311633, 1.78465117, 2.115

Predict

Cancrouse



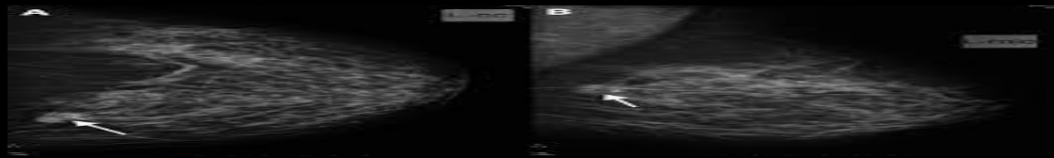
**Cancrouse**

Alert You have a Breast Cancer Disease, check yourself by doctor.

127.0.0.1:5000/predict

PROJECT online f... Home - Canva Responsive Our S... Overview | Wix.com Hdmovie2.com W... Breast\_Cancer\_Us... ABC | Academic B... All Bookmarks

## Breast Cancer Prediction Model (6861)




Input Breast Cancer Features

-0.23711093, -0.4976419, 0.61365274, -0.49813131, -0.53102815, -0.57694824, -0.17494424, -0.36215622, -0.284859, 0.43345165, 0.1781823

Predict

Not Cancrouse



**Not Cancrouse**

Don't Worry You don't have Breast Cancer Disease, Enjoy your life.

## **Conclusion:**

The cancer prediction system employs logistic regression to assess key features and predict the likelihood of cancer presence. Achieving an accuracy of approximately 97.37%, the model demonstrates high reliability and precision in generating predictions from input data. This robust performance highlights its potential as an effective tool for early cancer detection, aiding in timely diagnosis and intervention. The system's accuracy underscores its capability to support healthcare professionals in making informed decisions, ultimately contributing to improved patient outcomes and proactive healthcare management.