**Project Report**

**1.Introduction:**

**CancerVision: Advanced Breast Cancer Prediction With Deep Learning**

Breast cancer is one of the main causes of cancer death worldwide. Computer-aided diagnosis systems showed potential for improving the diagnostic accuracy. But early detection and prevention can significantly reduce the chances of death. It is important to detect breast cancer as early as possible. The goal is to classify images into two classifications of malignant and benign. As early diagnostics significantly increases the chances of correct treatment and survival. In this application we are helping the doctors and patients to classify the Type of Tumour for the specific image given with the help of Neural Networks.

2. **IDEATION & PROPOSED SOLUTION**

**2.1.  Problem Statement Definition**

Breast cancer is one of the main causes of cancer death worldwide. Early diagnostics significantly increases the chances of correct treatment and survival, but this process is tedious and often leads to a disagreement between pathologists.

**2.2 Empathy Map Canvas**

**What does she think and feel?**

Many people with cancer feel sad. They feel a sense of loss of their health, and the life they had before they learned they had the disease. Even when you're done with treatment, you may still feel sad. This is a common response to any serious illness.

**What does she see?**

Changes to your body shape, hair loss, early menopause and loss of fertility. Cancer can affect you and those close to you emotionally. Connecting with support services can help you to deal with these challenges.

**What does she hear?**

cancer patients want to hear that you already have a specific task in mind.

**What does she say and do?**

Don't ask what you can do to help or say, "Let me know if you need anything." Many people will never ask for help even though they need it.

**Pain:**

Pulling in of the nipple or pain in the nipple area. Nipple discharge other than breast milk, including blood. Any change in the size or the shape of the breast. Pain in any area of the breast.

**Gain:**

Medical procedures, prescription medicines, COBRA insurance coverage, housing expenses, utilities, transportation costs, and car insurance payments.

**2.3 Ideation & Brainstorming**

**Step-1: Problem Statement:**

Breast cancer is one of the main causes of cancer death worldwide. Early diagnostics significantly increases the chances of correct treatment and survival, but this process is tedious and often leads to a disagreement between pathologists.

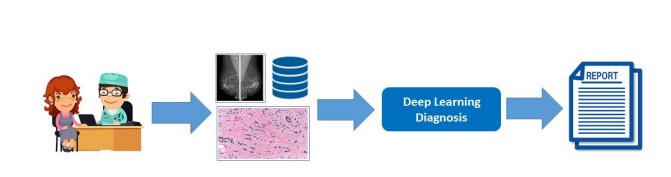
**Step-2: Idea listing:**

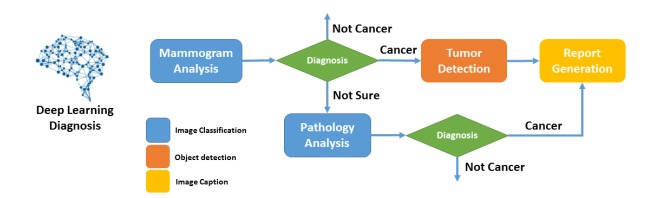
In recent times, different deep learning-based methods have been introduced for breast cancer diagnosis, which include CNN-, DNN-, RNN-, DBN- and AE-based approaches.

**Grouping:**

CNN is the most popular deep-learning technique that has been utilized in several studies for breast cancer detection

**Step-3: Idea Prioritization**

****

****

* **3. REQUIREMENT ANALYSIS**

**3.1 Functional requirement**

Following are the functional requirements of the proposed solution.

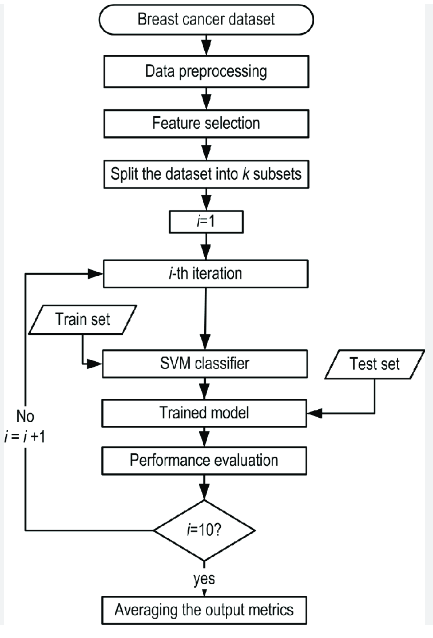
|  |  |  |
| --- | --- | --- |
| **FR No.** | **Functional Requirement (Epic)** | **Sub Requirement (Story / Sub-Task)** |
| FR-1 | User Image | Upload image through dataset |
| FR-2 | User values | Upload values through also dataset |

**3.2 Non-Functional requirements**

|  |  |  |
| --- | --- | --- |
| **FR No.** | **Non-Functional Requirement** | **Description** |
| NFR-1 | **Usability** | This application is very good to use and easy handling |
| NFR-2 | **Security** | It has an good security purposes |
| NFR-3 | **Reliability** | It has an no error has been given |
| NFR-4 | **Performance** | It gives high performance and quick reading |
| NFR-5 | **Availability** | It easily download through websites |

**4. PROJECT DESIGN**

**4.1 Data Flow Diagrams**

****

**4.2 User Stories**

| **User Type** | **Functional Requirement (Epic)** | **User Story Number** | **User Story / Task** | **Acceptance criteria** | **Priority** | **Team Member** |
| --- | --- | --- | --- | --- | --- | --- |
| Customer (use readings) | Readings | USN-1 | As a user, I can easily to know the status of breast cancer easily | I can check my status | High | Prithiga |
| Customer (application user) | login | USN-2 | As a user, I will surely conform the breast cancer is present or not | I can check cancer is present or not | High | pavin |
| Customer (image user) | image | USN-3 | As a user, I can easily upload my image and the reading to find the level of cancer | I can check while upload an image | Low | Sridhar |
| Customer (status user) | readings | USN-4 | As a user, it was easy and advanced process to check the status of cancer | I can check the status | Medium | Dharshini |

**5. CODING & SOLUTIONING**

# Check Python Version

import sys

import scipy

import numpy

import matplotlib

import pandas

import sklearn

print('Python: {}'.format(sys.version))

print('scipy: {}'.format(scipy.\_\_version\_\_))

print('numpy: {}'.format(numpy.\_\_version\_\_))

print('matplotlib: {}'.format(matplotlib.\_\_version\_\_))

print('pandas: {}'.format(pandas.\_\_version\_\_))

print('sklearn: {}'.format(sklearn.\_\_version\_\_))

Python: 3.10.11 (main, Apr 5 2023, 14:15:10) [GCC 9.4.0]

scipy: 1.10.1

numpy: 1.22.4

matplotlib: 3.7.1

pandas: 1.5.3

sklearn: 1.2.2

import numpy as np

from sklearn import preprocessing

from sklearn.model\_selection import train\_test\_split

from sklearn.neighbors import KNeighborsClassifier

from sklearn.svm import SVC

from sklearn import model\_selection

from sklearn.metrics import classification\_report

from sklearn.metrics import accuracy\_score

from pandas.plotting import scatter\_matrix

import matplotlib.pyplot as plt

import pandas as pd

# Load Dataset

url = "https://archive.ics.uci.edu/ml/machine-learning-databases/breast-cancer-wisconsin/breast-cancer-wisconsin.data"

names = ['id', 'clump\_thickness', 'uniform\_cell\_size', 'uniform\_cell\_shape',

       'marginal\_adhesion', 'single\_epithelial\_size', 'bare\_nuclei',

       'bland\_chromatin', 'normal\_nucleoli', 'mitoses', 'class']

df = pd.read\_csv(url, names=names)

# Preprocess the data

df.replace('?',-99999, inplace=True)

print(df.axes)

df.drop(['id'], 1, inplace=True)

[RangeIndex(start=0, stop=699, step=1), Index(['id', 'clump\_thickness', 'uniform\_cell\_size', 'uniform\_cell\_shape',

'marginal\_adhesion', 'single\_epithelial\_size', 'bare\_nuclei',

'bland\_chromatin', 'normal\_nucleoli', 'mitoses', 'class'],

dtype='object')]

<ipython-input-7-d9db8e418cb8>:5: FutureWarning: In a future version of pandas all arguments of DataFrame.drop except for the argument 'labels' will be keyword-only.

df.drop(['id'], 1, inplace=True)

# Let explore the dataset and do a few visualizations

print(df.loc[10])

# Print the shape of the dataset

print(df.shape)

clump\_thickness 1

uniform\_cell\_size 1

uniform\_cell\_shape 1

marginal\_adhesion 1

single\_epithelial\_size 1

bare\_nuclei 1

bland\_chromatin 3

normal\_nucleoli 1

mitoses 1

class 2

Name: 10, dtype: object

(699, 10)

# Describe the dataset

print(df.describe())

clump\_thickness uniform\_cell\_size uniform\_cell\_shape \

count 699.000000 699.000000 699.000000

mean 4.417740 3.134478 3.207439

std 2.815741 3.051459 2.971913

min 1.000000 1.000000 1.000000

25% 2.000000 1.000000 1.000000

50% 4.000000 1.000000 1.000000

75% 6.000000 5.000000 5.000000

max 10.000000 10.000000 10.000000

marginal\_adhesion single\_epithelial\_size bland\_chromatin \

count 699.000000 699.000000 699.000000

mean 2.806867 3.216023 3.437768

std 2.855379 2.214300 2.438364

min 1.000000 1.000000 1.000000

25% 1.000000 2.000000 2.000000

50% 1.000000 2.000000 3.000000

75% 4.000000 4.000000 5.000000

max 10.000000 10.000000 10.000000

normal\_nucleoli mitoses class

count 699.000000 699.000000 699.000000

mean 2.866953 1.589413 2.689557

std 3.053634 1.715078 0.951273

min 1.000000 1.000000 2.000000

25% 1.000000 1.000000 2.000000

50% 1.000000 1.000000 2.000000

75% 4.000000 1.000000 4.000000

max 10.000000 10.000000 4.000000

# Plot histograms for each variable

df.hist(figsize = (5, 10))

plt.show()

# Create scatter plot matrix

scatter\_matrix(df, figsize = (20,20))

plt.show()

# Create X and Y datasets for training

X = np.array(df.drop(['class'], 1))

y = np.array(df['class'])

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2

<ipython-input-13-7d55bc2ba08d>:2: FutureWarning: In a future version of pandas all arguments of DataFrame.drop except for the argument 'labels' will be keyword-only.

X = np.array(df.drop(['class'], 1))

# Testing Options

seed = 8

scoring = 'accuracy'

# Make predictions on validation dataset

for name, model in models:

    model.fit(X\_train, y\_train)

    predictions = model.predict(X\_test)

    print(name)

    print(accuracy\_score(y\_test, predictions))

    print(classification\_report(y\_test, predictions))

# Accuracy - ratio of correctly predicted observation to the total observations.

# Precision - (false positives) ratio of correctly predicted positive observations to the total predicted positive observations

# Recall (Sensitivity) - (false negatives) ratio of correctly predicted positive observations to the all observations in actual class - yes.

# F1 score - F1 Score is the weighted average of Precision and Recall. Therefore, this score takes both false positives and false

# Make predictions on validation dataset

for name, model in models:

    model.fit(X\_train, y\_train)

    predictions = model.predict(X\_test)

    print(name)

    print(accuracy\_score(y\_test, predictions))

    print(classification\_report(y\_test, predictions))

# Accuracy - ratio of correctly predicted observation to the total observations.

# Precision - (false positives) ratio of correctly predicted positive observations to the total predicted positive observations

# Recall (Sensitivity) - (false negatives) ratio of correctly predicted positive observations to the all observations in actual class - yes.

# F1 score - F1 Score is the weighted average of Precision and Recall. Therefore, this score takes both false positives and false

clf = SVC()

clf.fit(X\_train, y\_train)

accuracy = clf.score(X\_test, y\_test)

print(accuracy)

example\_measures = np.array([[4,2,1,1,1,2,3,2,1]])

example\_measures = example\_measures.reshape(len(example\_measures), -1)

prediction = clf.predict(example\_measures)

print(prediction)

0.5357142857142857

[2]

**6. RESULTS**

It gives an accurate readings