Breast Cancer Detection Using Machine Learning

Somya Goyal, Mehul Sinha, Shashwat Nath, Sayan Mitra, and Charvi Arora

Abstract Through studies and statistics, it has been found that these days, breast cancer is the most common cancer leading to frequent deaths among women. Early screening and subsequent treatment can raise the chances of survival. Through this paper, we aim to demonstrate the ability to detect breast cancer cases using MRI scan data by analyzing the given data with machine learning algorithms. Using machine learning, we hope to ease the process of cancer detection in the hospitals so the patient can be afforded the right treatment as soon as possible before the situation can become critical. It also opens the door toward new possibilities in cancer detection for other different types of cancers as well as other diseases by use of machine learning in medical science, where detection using conventional means is usually laborious and time-taking.

1 Introduction

During the past years, doctors have classified breast cancer into various subtypes. In 2020, there were 2.3 million women diagnosed with breast cancer and 685,000 deaths globally. At the end of 2020, there were 7.8 million women alive who were diagnosed with breast cancer in the past 5 years. According to GLOBOCAN, breast cancer is the predominant cancer in women, making up to 25.1% of all cancers. There has been a significant decline in the amount of cases of breast cancer because of improvement in the medical sector with the help of advanced technology and software. Over a period of time, machine learning has helped a lot in improving the accuracy of detection of breast cancer in early stages. Breast cancer is more difficult and costly to treat as it reaches upper stages, hence, proper machinery and technology for detection and treatment are called for. This paper lays out a machine learning algorithm that helps to detect breast cancer in women efficiently and with great accuracy.

S. Goyal (✉) · M. Sinha · S. Nath · S. Mitra · C. Arora
Manipal University Jaipur, Jaipur, Rajasthan 303007, India
e-mail: somyagoyal1988@gmail.com

V. Bhateja et al. (eds.), *Communication, Software and Networks*, Lecture Notes in Networks and Systems 493, https://doi.org/10.1007/978-981-19-4990-6_57

2 Related Works

This section gives the review of literature work in the field of "Breast cancer detection using machine learning." Multiple sources were reviewed by the authors for the literature which have been referred for the analysis of breast cancer using machine learning. Further, the authors reviewed datasets from reliable sources for testing purposes of the methods reviewed. The most popular methods [1] of breast cancer detection, namely Naive Bayes classifier, K-nearest neighbors (KNN) [2], logistic regression, decision tree classifier, support vector machine (SVM) classifier [3] and random forest classifier [4, 5], which are given in Table 1.

3 Proposed Methodology

The purpose of this work is to analyze the dataset using random forest classifier, logistic regression and decision tree classifier and compare the efficiency of the respective algorithms in terms of accuracy in detecting cancer in the subject.

3.1 Experimental Setup

The program was made using Python programming language on Jupyter Notebook application. We have used the Wisconsin Breast Cancer datasets [14] from the UCI machine learning repository for our analysis regarding the scope of our work.

We imported NumPy library for working with data using arrays. We imported Pandas library to analyze, clean and manipulate the given dataset. We imported Matplotlib and Seaborn libraries for data visualization of the statistics. We made use of Sclearn which is a machine learning library for Python to import and implement logistic regression, random forest classifier and decision tree classifier algorithms on the given data.

Logistic Regression: It uses a logistic function to for modeling data using dependent and independent variable. It is used for binary data and it is efficient to train. Logistic regression algorithm performs efficiently when the given dataset is separable linearly.

Decision Tree Classifier: A decision tree consists of decision nodes and leaf nodes. It begins with the root node and finds the best attribute using attribute selection measure so as to divide it into multiple subsets. From these, a decision tree node is generated. The process repeats recursively until no further classification of nodes is possible. It makes predictions from all types of outcomes.

Random Forest Classifier: It uses the functionality of a group of decision trees. It is an ensemble algorithm. Individual trees are generated by attribute selection indication. Greater number of trees lead to computation of a better average and thus

Table 1 Literature work

S. No	Study	Technique used	Strength	Weakness
1	Tahmooresi et al. [6]	Artificial neural network (ANN) Support vector machine (SVM) K-nearest neighbors (KNN)	SVM is a performant algorithm and produces accurate results	Large datasets not suitable for SVM algorithm
2	Nallamala et al. [7]	NumPy Pandas	Powerful libraries for representation and manipulation of data	None
3	Bazazeh et al. [8]	Random forest (RF) Bayesian networks (BN)	Stable and unbiased algorithm producing accurate results	Heavy use of computational resources
4	Agarap et al [9]	GRU-SVM Linear regression	Used for predictive analysis	Performs poorly in nonlinear relationships between variables
5	Vaka, Soni et al. [10]	R-CNN (convolutional neural networks) classifier Bidirectional recurrent neural networks (HA-BiRNN)	Deep learning models with unsupervised feature extraction	Computationally taxing and big dataset required for training
6	Chaurasia et al. [11]	Sequential minimal optimization (SMO) BF tree	Resolves quadratic programming problem that arises during training of SVM	Necessary to solve QP scaling with number of SVMs
7	Gayathri et al. [12]	Supervised, unsupervised, semi-supervised and reinforcement learning	Optimize performance in case of supervised algorithm and implement real-world problems	Preprocessing required. Computation time is vast for supervised algorithms
8	Ganggayah et al. [13]	Support vector machine neural networks logistic regression	Outperforms most of the machine learning algorithms	Unpredictable output and computationally its very expensive

increases the accuracy of the making predictions on the given dataset. It resolves the overfitting problem that arises with decision trees.

3.2 Feature Analysis

First up, the features are analyzed for their frequency count [15], and then, their pairwise correlation is computed. The code is given below.

```
In [1]: #This program detects breast cancer using data.

In [2]: #import libraries
        import numpy as np
        import pandas as pd
        import matplotlib.pyplot as plt
        import seaborn as sns

In [3]: #Load the data
        df = pd.read_csv('data.csv')

In [4]: #count the number of rows and coloumns
        df.shape
Out[4]: (569, 32)
```

The plot for the count of diagnosis for malignant (M) and benign (B) is shown in Fig. 1. The pairwise correlation is shown with the help of a heat map [15–17] as in Fig. 2.

```
In [8]: #get a count of Malignant (M) or Begnin(B) cells
        df['diagnosis'].value_counts()
Out[8]: B    357
        M    212
        Name: diagnosis, dtype: int64

In [9]: #visualize the count
        sns.countplot(df['diagnosis'], label = 'count')
```

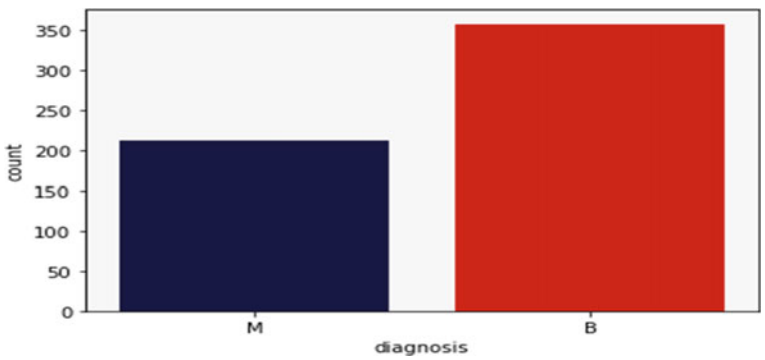

Fig. 1 Frequency plot for two classes-malignant (M) and benign (B)

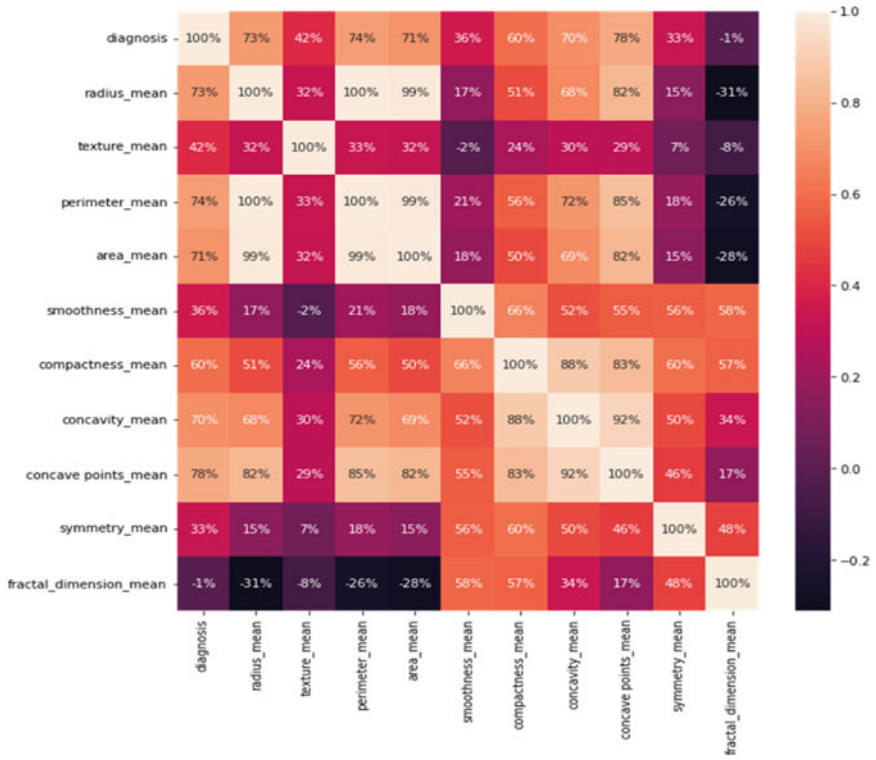

Fig. 2 Heat map plot for pairwise correlation among features

```
In [11]: #Encode the catogorical data values
         from sklearn.preprocessing import LabelEncoder
         labelencoder_Y = LabelEncoder()
         df.iloc[:,1] = labelencoder_Y.fit_transform(df.iloc[:,1].values)

In [12]: #Create a pair plot
         sns.pairplot(df.iloc[:,1:8], hue = 'diagnosis')
```

```
In [13]: #get the correlation of the cloumns
         df.iloc[:, 1:12].corr()
```

Out[39]:

	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean	concavity_mean
diagnosis	1.000000	0.730029	0.415185	0.742636	0.708984	0.358560	0.596534	0.696360
radius_mean	0.730029	1.000000	0.323782	0.997855	0.987357	0.170581	0.506124	0.676764
texture_mean	0.415185	0.323782	1.000000	0.329533	0.321086	-0.023389	0.236702	0.302418
perimeter_mean	0.742636	0.997855	0.329533	1.000000	0.986507	0.207278	0.556936	0.716136
area_mean	0.708984	0.987357	0.321086	0.986507	1.000000	0.177028	0.498502	0.685983
smoothness_mean	0.358560	0.170581	-0.023389	0.207278	0.177028	1.000000	0.659123	0.521984
compactness_mean	0.596534	0.506124	0.236702	0.556936	0.498502	0.659123	1.000000	0.883121
concavity_mean	0.696360	0.676764	0.302418	0.716136	0.685983	0.521984	0.883121	1.000000
concave points_mean	0.776614	0.822529	0.293464	0.850977	0.823269	0.553695	0.831135	0.921391
symmetry_mean	0.330499	0.147741	0.071401	0.183027	0.151293	0.557775	0.602641	0.500667

```
In [14]: #Visualize the correlation
         plt.figure(figsize = (10,10))
         sns.heatmap(df.iloc[:, 1:12].corr(), annot = True, fmt='.0%' )
```

The classification models are constructed. After training the model with the given algorithms, we computed accuracy and confusion matrix from Sclearn library to ascertain the performance of the algorithms on the constructed model.

```
In [15]: #split the data set into independent (X) and dependent (Y) data sets
         X = df.iloc[:, 2:31].values
         Y = df.iloc[:, 1].values
```

```
In [16]: #split the data set into 80% training and 20% testing
         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, random_state=0)
```

```
In [17]: #Scale the data (Feature Scaling)
         from sklearn.preprocessing import StandardScaler
         sc = StandardScaler()
         X_train = sc.fit_transform(X_train)
         X_test = sc.fit_transform(X_test)
```

```
In [18]: #Create a function for the models
         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(criterion= 'entropy' , random_state = 0)
           tree.fit(X_train, Y_train)

           #Random forest classifier
           from sklearn.ensemble import RandomForestClassifier
           forest = RandomForestClassifier(n_estimators = 10, criterion = 'entropy', random_state = 0)
           forest.fit(X_train, Y_train)

           #Print the models accuracy on the training data
           print('Logistic Regression gives a Training Accuracy ', log.score(X_train, Y_train))
           print('Decision Tree gives a Training Accuracy ', tree.score(X_train, Y_train))
           print('Random Forest Classifier gives a Training Accuracy ', forest.score(X_train, Y_train))

           return log, tree, forest

In [19]: #Getting all of the models
         model = models(X_train, Y_train)
```

We calculated the accuracy of each model using a confusion matrix:

TrPo True Positive
TrNe True Negative
FaNe False Negative
FaPo False positive

$$\text{Accuracy} = (\text{TrPo} + \text{TrNe})/(\text{TrPo} + \text{TrNe} + \text{FaNe} + \text{FaPo})$$

```
In [20]: #test model accuracy on test data on confussion matrix
         from sklearn.metrics import confusion_matrix

         for i in range(len(model)):
           print('Model ', i)
           cm = confusion_matrix(Y_test, model[i].predict(X_test))
           TP = cm[0][0]
           TN = cm[1][1]
           FN = cm[1][0]
           FP = cm[0][1]

           print(cm)
           print('testing accuracy =',( TP + TN ) / ( TP + TN + FN + FP ))
           print()
```

Fig. 3 Confusion matrix for logistic regression

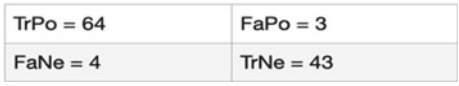

Fig. 4 Confusion matrix for decision tree

Fig. 5 Confusion matrix for random forest classifier

4 Results and Discussions

We were successful in detection of breast cancer in patients through the use of machine learning algorithms. To assess the performance of all the algorithms, we used a confusion matrix to provide the best evaluation. We took 80% of the dataset to train each of the model and 20% of the dataset to test the precision of each model.

Logistic regression algorithm has an accuracy of 96.49%, while the decision tree algorithm has an accuracy of only 93.85%. However, the best results were shown by random forest classifier algorithm with an accuracy of 97.36%. Hence, we are able to tell with high accuracy if the cancer is benign or malignant in the subject (Figs. 3, 4 and 5).

5 Conclusions

Breast cancer is one of the most predominant cancers found in women. Detection at an early stage and diagnosis of this disease can save lives and the patient can undergo suitable treatment. Machine learning has vast applications in the modern healthcare system. One such use of machine learning is detection and diagnosis of diseases that has been thoroughly discussed in this paper. Integration of machine learning with medical databases and devices will make healthcare system more efficient and help in better organization of data. Healthcare industries such as tempus are using machine learning on their clinical data to provide personalized treatments for patients. Hence, an increased adoption of machine learning technology is expected in medical fields in the future.

References

1. Goyal S, Bhatia PK (2020) Comparison of machine learning techniques for software quality prediction. Int J Knowl Syst Sci (IJKSS) 11(2):21–40. https://doi.org/10.4018/IJKSS.2020040102
2. Goyal S (2021) Handling class-imbalance with KNN (Neighbourhood) under-sampling for software defect prediction. Artif Intell Rev. https://doi.org/10.1007/s10462-021-10044-w
3. Goyal S (2021) Effective software defect prediction using support vector machines (SVMs). Int J Syst Assur Eng Manag. https://doi.org/10.1007/s13198-021-01326-1
4. Goyal S, Bhatia PK (2021) Heterogeneous stacked ensemble classifier for software defect prediction. Multimed Tools Appl. https://doi.org/10.1007/s11042-021-11488-6
5. Goyal S (2021) Predicting the defects using stacked ensemble learner with filtered dataset. Autom Softw Eng 28:14. https://doi.org/10.1007/s10515-021-00285-y
6. Tahmooresi M, Afshar A, Rad BB, Nowshath KB, Bamiah MA (2018) Early detection of breast cancer using machine learning techniques. J Telecommun Electr Comput Eng (JTEC) 10(3–2):21–27
7. Nallamala SH, Mishra P, Koneru SV (2019) Breast cancer detection using machine learning way. Int J Recent Technol Eng 8:1402–1405
8. Bazazeh D, Shubair R (2016) Comparative study of machine learning algorithms for breast cancer detection and diagnosis. In: 2016 5th international conference on electronic devices, systems and applications (ICEDSA). IEEE, Dec 2016, pp 1–4
9. Agarap AFM (2018) On breast cancer detection: an application of machine learning algorithms on the wisconsin diagnostic dataset. In: Proceedings of the 2nd international conference on machine learning and soft computing, Feb 2018, pp 5–9
10. Vaka AR, Soni B, Reddy S (2020) Breast cancer detection by leveraging machine learning. ICT Express 6(4):320–324
11. Chaurasia V, Pal S (2017) A novel approach for breast cancer detection using data mining techniques. Int J Innov Res Comput Commun Eng 2 (An ISO 3297: 2007 Certified Organization)
12. Gayathri BM, Sumathi CP, Santhanam T (2013) Breast cancer diagnosis using machine learning algorithms-a survey. Int J Distrib Parallel Syst 4(3):105
13. Ganggayah MD, Taib NA, Har YC, Lio P, Dhillon SK (2019) Predicting factors for survival of breast cancer patients using machine learning techniques. BMC Med Inform Decis Mak 19(1):1–17
14. https://www.kaggle.com/uciml/breast-cancer-wisconsin-data/version/2
15. Goyal S, Bhatia PK (2021) Software fault prediction using lion optimization algorithm. Int J Inf Tecnol. https://doi.org/10.1007/s41870-021-00804-w
16. Goyal S, Bhatia PK (2020) Feature selection technique for effective software effort estimation using multi-layer perceptrons. In: Proceedings of ICETIT 2019. Lecture Notes in Electrical Engineering, vol 605. pp 183–194. Springer, Cham. https://doi.org/10.1007/978-3-030-30577-2_15
17. Goyal S (2022) FOFS: firefly optimization for feature selection to predict fault-prone software modules. In: Nanda P, Verma VK, Srivastava S, Gupta RK, Mazumdar AP (eds) Data engineering for smart systems. Lecture Notes in Networks and Systems, vol 238. Springer, Singapore. https://doi.org/10.1007/978-981-16-2641-8_46