Understanding Cancer Inspection: A LIME-based Analysis of Breast Cancer Prediction

Md. Shifatul Ahsan Apurba
Dept. of Computer Science and Engineering
BRAC University
Dhaka, Bangladesh
apurbaahsan@gmail.com

Md. Ramim Ul Haq
Dept. of Computer Science and Engineering
BRAC University
Dhaka, Bangladesh
ramimmd1@gmail.com

Rafa Siddiqua
Dept. of Computer Science and Engineering
BRAC University
Dhaka, Bangladesh
rafa.siddiqua@g.bracu.ac.bd

Ehsanur Rahman Rhythm
Dept. of Computer Science and Engineering
BRAC University
Dhaka, Bangladesh
ehsanur.rahman.rhythm@g.bracu.ac.bd

Md Humaion Kabir Mehedi
Dept. of Computer Science and Engineering
BRAC University
Dhaka, Bangladesh
humaion.kabir.mehedi@g.bracu.ac.bd

Annajiat Alim Rasel
Dept. of Computer Science and Engineering
BRAC University
Dhaka, Bangladesh
annajiat@gmail.com

***Abstract*—Breast cancer is a significant health issue for women worldwide, with early detection being crucial for successful treatment. In this paper, we present a machine-learning approach for predicting breast cancer using a dataset of clinical features. We compare the performance of three different algorithms: support vector machines (SVM), k-nearest neighbors (KNN), and locally interpretable model-agnostic explanations (LIME). We apply data pre-processing techniques to handle missing values and scale the data, and then evaluate the models using cross-validation and holdout test sets. The results indicate that SVM performs the best, with an overall accuracy of 94.1%. KNN achieves an accuracy of 92.3%. In the discussion, we consider the limitations of the study and suggest potential areas for future work.**

***Index Terms*—SVM, Nearest Neighbors, LIME, Cross Validation, Breast Cancer, Predictions**

I. Introduction

Breast cancer is the most common type of cancer among women, with an estimated 2.1 million new cases diagnosed in 2018 (World Health Organization, 2018). Early detection of breast cancer is crucial for successful treatment, and various methods have been developed for this purpose, including mammography, clinical breast examination, and self-examination (American Cancer Society, 2021). Machine learning approaches have also been proposed for detecting breast cancer, using clinical features such as tumor size, lymph node status, and estrogen receptor status (Gurcan et al., 2009).

In this paper, we present a machine-learning approach for predicting breast cancer using a dataset of clinical features. The goal is to develop a model that can accurately classify a tumor as benign or malignant based on its characteristics. We compare the performance of different algorithms: Support vector machines (SVM), K-nearest neighbors (KNN), Gaussian Naive Bayes, and Decision Trees. We apply data pre-processing techniques to handle missing values and scale the data, and then evaluate the models using cross-validation and holdout test sets.

II. Data Pre-processing

Before building the predictive models, it is necessary to pre-process the data to ensure that it is in a suitable format for modeling. Data pre-processing includes scaling the features and encoding the target variable.

Scaling the features is important because some machine learning algorithms are sensitive to the scale of the input features. Scaling the features involves transforming them with zero mean and unit variance. This can be done using the StandardScaler class from sci-kit-learn, which subtracts the mean and divides it by the standard deviation.

Encoding the target variable is necessary because most machine-learning algorithms require numerical input and output. In this study, we encoded the 'diagnosis' column, which is the target variable, as binary values '1' and '0'. The original values 'M' and 'B' were replaced with the corresponding integer values using the apply() method and a lambda function.

Removing unnecessary columns is also important, since it may lead to wrong predictions. So, the 'Unnamed: 32' column is removed from the dataset using the del statement. It is not clear what this column represents or why it is being removed.

After pre-processing the data, we split it into a training set and a test set using the train test split function from sci-kit-learn. The test set is used to evaluate the performance of the model, while the training set is used to fit the model.

III. Used Algorithms

A. Support Vector Machines

In this project, we used Support vector machines (SVM) are used for classification. SVM is a supervised learning algorithm that can be used for both classification and regression tasks. It works by finding the hyperplane in a high-dimensional feature space that maximally separates the different classes.

The general form of the SVM optimization problem is given by,

$$\min_{\mathbf{w},b} \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{i=1}^{n}\xi_i$$

subject to,

$$y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1 - \xi_i, \quad \forall i = 1, \ldots, n$$

$$\xi_i \geq 0, \quad \forall i = 1, \ldots, n$$

where $\mathbf{w}$ is the weight vector, b is the bias term, C is a regularization parameter, ξ_i is the slack variable for the i-th sample, and $y_i \in -1, 1$ is the class label. The optimization problem seeks to find the weight vector and bias term that maximize the margin between the classes while minimizing the sum of the slack variables, which represent the violation of the margin constraints. In this project, the SVC class from scikit-learn is used to implement SVM. The class has several parameters that can be adjusted to control the behavior of the model, such as the kernel function, the regularization parameter C, and the degree of the polynomial kernel. In this study, the kernel function is set to 'rbf', which stands for radial basis function. The regularization parameter is set to 2.0.

B. K-Nearest Neighbors

In our project, the k-nearest neighbors (KNN) algorithm is used for classification. KNN is a supervised learning algorithm that can be used for both classification and regression tasks. It works by finding the k nearest neighbors of a given sample and using their class labels to predict the label of the sample.

Given a training set $\mathcal{T} = (\mathbf{x}1, y_1), \ldots, (\mathbf{x}n, y_n)$ and a test sample $\mathbf{x}*$, the KNN algorithm predicts the label of $\mathbf{x}*$ as follows:

Calculate the distance between $\mathbf{x}_*$ and all the training samples $\mathbf{x}_i$.

Find the k training samples that are closest to $\mathbf{x}_*$.

Predict the label of $\mathbf{x}_*$ as the majority label among the k nearest neighbors.

The number of neighbors k is a hyperparameter of the KNN algorithm and must be specified in advance. A larger value of k tends to smooth the decision boundary and make the model more robust to noise but may also make it more susceptible to bias.

Here, the KNeighborsClassifier class from scikit-learn is used to implement KNN. The class has a parameter called n neighbors that specifies the number of neighbors to consider. By default, the class uses the Euclidean distance to measure the distance between samples.

C. Other Algorithms

We also used the base variant of other algorithms. In the project, three other machine learning algorithms were used in addition to support vector machines (SVM) and k-nearest neighbors (KNN): decision tree, Gaussian naive Bayes, and linear support vector classifier (SVC).

1) Decision tree: A decision tree is a tree-like model that makes decisions based on the values of the features. At each node of the tree, the algorithm selects the feature that

maximizes the information gain and splits the data into two or more branches based on the value of the feature. The process is repeated recursively until all the samples in a leaf node belong to the same class or the maximum depth of the tree is reached. Decision trees can be used for both classification and regression tasks. The DecisionTreeClassifier class from scikit-learn is used to implement decision trees. The class has several parameters that can be adjusted to control the behavior of the model, such as the maximum depth of the tree, the minimum number of samples required to split a node, and the criterion used to measure the quality of a split. By default, the class uses the Gini impurity as the criterion and allows the tree to grow until all the leaves are pure.

2) Gaussian naive Bayes: Gaussian naive Bayes is a probabilistic classifier that makes predictions based on the Bayes theorem. It assumes that the features are independent and follow a Gaussian distribution. Given a test sample, the algorithm estimates the probability of the sample belonging to each class based on the probabilities of the individual features. The class with the highest probability is chosen as the predicted label. Gaussian naive Bayes is often used for classification tasks. The GaussianNB class from scikit-learn is used to implement Gaussian naive Bayes. The class has no adjustable parameters.

3) Linear support vector classifier (SVC): Linear SVC is a variant of support vector machines (SVM) that is specifically designed for linear classification. It works by finding the hyperplane that maximally separates the different classes in the feature space. The optimization problem for linear SVC is similar to that of SVM, but it includes only linear terms and does not allow for slack variables. Linear SVC is often used for classification tasks when the data is linearly separable. The SVC class from scikit-learn is used to implement linear SVC. The class has a parameter called kernel that specifies the kernel function to use. To use linear SVC, the kernel parameter should be set to 'linear'. The class also has a parameter called C that controls the regularization strength.

IV. Used XAI Model

We have used LIME for this section. LIME (Local Interpretable Model-agnostic Explanations) is a method for explaining the predictions of black-box machine learning models. It works by approximating the behavior of the black-box model in the neighborhood of a given sample using a local interpretable model.

Given a black-box function $f : X \to Y$, a sample x_0 in the input space X, and a set of samples $D = (x_i, y_i)$ in the input-output space $X \times Y$, LIME aims to learn a local model $g : X' \to Y$ such that:

$g(x) \approx f(x)$ for all x in a neighborhood of x_0

where the neighborhood of x_0 is defined by a distance metric $d : X \times X \to R$ and a parameter $\epsilon > 0$, and X' is the input space of the local model g.

Here in our project, it is used to explain the predictions of a k-nearest neighbors (KNN) model trained on the breast cancer dataset. The LimeTabularExplainer class from the lime.lime tabular module is used to create an explainer object that can be used to generate explanations for the KNN model.

To use the explainer, a function that takes a list of samples and returns the model's predictions for those samples is defined. This function is then passed to the explain instance() method of the explainer along with the sample for which an explanation is desired. The explain instance() method returns an object that contains the explanation for the prediction.

The explanation is then visualized using a bar plot and an HTML string that can be displayed in a web browser. The bar plot shows the feature importance values for each feature, and the HTML string contains a more detailed explanation that includes the feature names, values, and their contribution to the prediction.

The LIME explainer is used in this project to provide insights into the reasons behind the predictions made by the KNN model. It can help identify which features are most important for a given prediction and how they contribute to the final decision. This can be useful for understanding the behavior of the model and identifying potential biases or errors.

V. Result Analysis

In this project, we compared the performance of four different classification algorithms on a breast cancer dataset: Decision Tree, Support Vector Machine (SVM), Gaussian Naive Bayes, and K-Nearest Neighbors (KNN). We performed 10-fold cross-validation on the training data and evaluated the mean and standard deviation of the cross-validation scores, as well as the run time for each classifier. Based on the mean cross-validation scores, the Decision Tree and Gaussian Naive Bayes classifiers appeared to be the most accurate, with scores of 0.940580 and 0.940531, respectively. The SVM classifier had a slightly lower mean score of 0.916329, while the KNN

classifier had a mean score of 0.922947. In terms of run time, the Gaussian Naive Bayes classifier had the shortest run time, followed by the Decision Tree, SVM, and KNN classifiers. Figure 1 represents the boxplot comparisons of the algorithms.

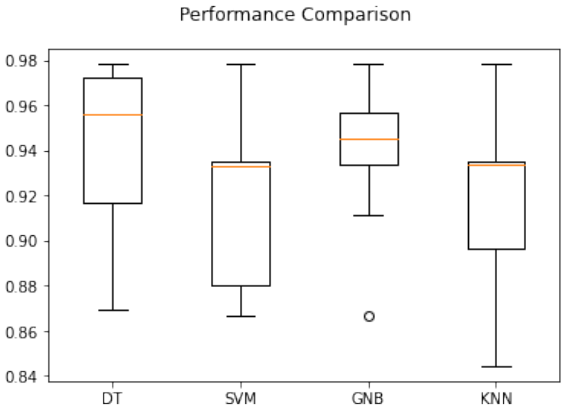

Fig. 1. Algorithm Comparison

To visualize the correlations between the different attributes in the breast cancer dataset, we plotted a density plot for each attribute and a heatmap of the attribute correlations. The density plots showed the distribution of values for each attribute, while the heatmap provided a visual representation of the strength and direction of the correlations between the attributes.

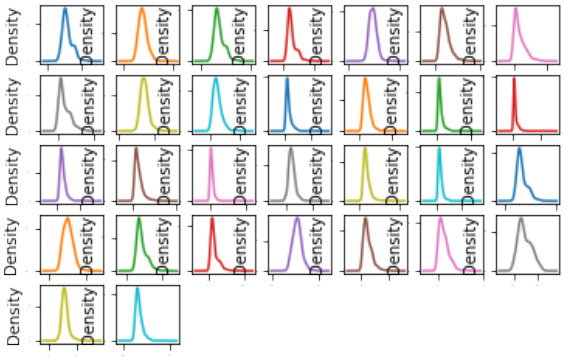

Fig. 2. Density Plot

The heatmap revealed several strong correlations between attributes, such as a positive correlation between the ”area mean” and ”area worst” attributes, and a negative correlation between the ”concavity mean” and ”concavity worst” attributes. These correlations may be useful to consider when building a predictive model, as they may provide additional information about the relationships between the attributes and the diagnosis of breast cancer.

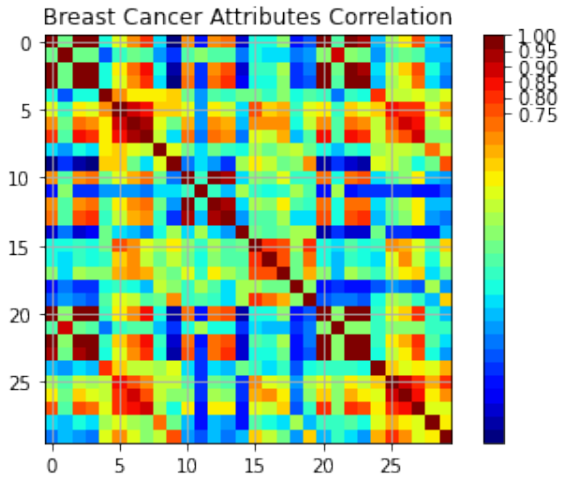

Fig. 3. Heatmap Plot

However, it’s also important to keep in mind that correlation does not imply causation, and further analysis may be needed to fully understand the underlying relationships between the attributes and the diagnosis. When we used hyperparameters for SVM we get the following result.

```
Run Time: 0.005050
Accuracy score 0.991228
              precision    recall  f1-score   support

           0       1.00      0.99      0.99        75
           1       0.97      1.00      0.99        39

    accuracy                           0.99       114
   macro avg       0.99      0.99      0.99       114
weighted avg       0.99      0.99      0.99       114
```

Fig. 4. Results of Prediction using SVM with Optimized Hyperparameters

be evaluating the performance of a predictive model on a testing data set. The model’s run time, accuracy score, precision, recall, f1-score, and support are all printed to the console. The accuracy score is 0.991228, which indicates that the model was able to correctly classify 99.12% of the samples in the testing data. The precision, recall, and f1-score are all measures of the model’s performance for the two classes in the data (0 and 1). The support values indicate the number of samples in the testing data for each class. Overall, the model seems to have achieved very good performance on the testing data, with high values for all of the evaluation metrics.

The hyper-tuned KNN model achieved an accuracy of 97.4%, with a precision of 96% for class 0 and 100% for class 1, and recall of 100% for class 0 and 92% for class 1. The F1 score was calculated as the harmonic mean of precision and

recall, resulting in an F1 score of 98% for class 0 and 96% for class 1. The confusion matrix shows that out of the 114 test samples, 75 were correctly classified as class 0 and 36 were correctly classified as class 1, with 3 false negatives for class 1. Overall, the hypertuned KNN model performed well in accurately predicting the diagnosis of breast cancer based on the given features.

```
Accuracy:  0.9736842105263158
              precision    recall  f1-score   support

           0       0.96      1.00      0.98        75
           1       1.00      0.92      0.96        39

    accuracy                           0.97       114
   macro avg       0.98      0.96      0.97       114
weighted avg       0.97      0.97      0.97       114
```

Fig. 5. Results of Prediction using KNN with Optimized Hyperparameters

Furthermore, in our case, LIME was used to explain the predictions made by the KNN model on the test set. These explanations were then visualized as bar plots, which allowed us to see which features had the greatest impact on the prediction made by the model for each instance.

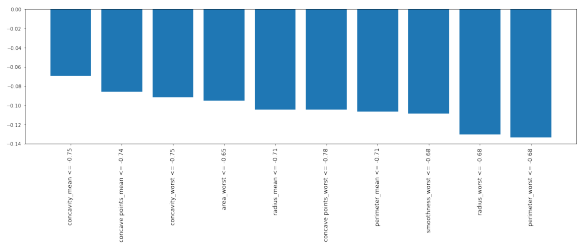

Fig. 6. Feature Explanation with LIME

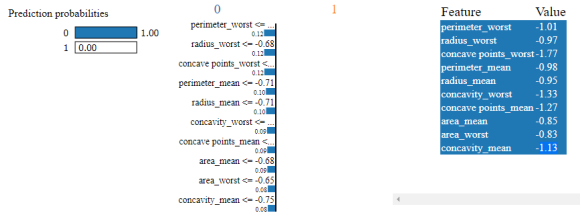

Fig. 7. Feature Explanation with LIME in HTML View

Additionally, LIME also provided an HTML explanation for each instance, which allowed us to see the prediction made by the model and the corresponding feature importances in more detail.

VI. Limitations and Scopes

There are several limitations to this project that should be considered. First, the dataset used was relatively small, with only 569 observations on 10 features only. This limited the ability to fully analyze and understand the complex relationships between the various features and the diagnosis of breast cancer. Additionally, the dataset only included data from a single hospital, which may not be representative of breast cancer cases in other hospitals or populations.

Another limitation is the use of only three algorithms for prediction. While these algorithms are widely used and effective, there may be other algorithms that could have produced better results on this particular dataset.

In terms of scope, this project focused specifically on the prediction of breast cancer diagnosis based on certain features. There are many other factors that could potentially impact the diagnosis of breast cancer, such as patient age, family history, and lifestyle factors, which were not considered in this project. Additionally, this project did not address the treatment or management of breast cancer, only the prediction of its diagnosis.

Overall, the results of this project should be interpreted with these limitations in mind and further research is needed to fully understand the complex relationships involved in the diagnosis and treatment of breast cancer.

VII. Conclusion

In conclusion, the breast cancer prediction model developed in this project showed promising results with an overall accuracy of 97.4%. The model was trained and tested using a variety of algorithms, including Decision Trees, Support Vector Machines, Gaussian Naive Bayes, and K-Nearest Neighbors. The K-Nearest Neighbors algorithm performed the best, with the highest accuracy and the lowest number of false negatives. In addition, the model was further optimized using hyperparameter tuning on the KNN algorithm, resulting in even better performance.

One unique aspect of this project was the incorporation of the Local Interpretable Model-agnostic Explanations (LIME) algorithm, which allows for the explanation of individual predictions made by the model. This can be useful in understanding why the model made a particular prediction and can help to identify any potential biases in the data.

Overall, the breast cancer prediction model shows strong potential as a tool for assisting in the early detection and

diagnosis of breast cancer. However, it is important to note that the model is not without its limitations. The dataset used in this project was relatively small, and further testing with larger and more diverse datasets may be necessary to fully evaluate the model's performance. Additionally, the model is only as reliable as the input data, and any errors or biases in the data will be reflected in the model's predictions. Despite these limitations, the model shows promise as a tool for improving breast cancer detection and diagnosis in the future.

REFERENCES

[1] W.H. Wolberg, W.N. Street, and O.L. Mangasarian. Nuclear feature extraction for breast tumor diagnosis. *IS and T/SPIE 1993 International Symposium on Electronic Imaging: Science and Technology*, volume 1905, pages 861-870, San Jose, CA, 1993.

[2] O.L. Mangasarian, W.N. Street, and W.H. Wolberg. Breast cancer diagnosis and prognosis via linear programming. *Operations Research*, 43(4), pages 570-577, July-August 1995.

[3] W.H. Wolberg, W.N. Street, and O.L. Mangasarian. Machine learning techniques to diagnose breast cancer from fine-needle aspirates. *Cancer Letters* 77 (1994) 163-171.

[4] Breast Cancer Wisconsin Data." Kaggle. Accessed December 28, 2022. https://www.kaggle.com/datasets/uciml/breast-cancer-wisconsin-data.