Lung Cancer Detection Using Classification Algorithms

Abdullah Muhammad Tahir
ID: 2017-1-60-082
Computer Science & Engineering
East West University
Dhaka, Bangladesh
Email: 2017-1-60-082@std.ewubd.edu

Tanvir Ahmed
ID: 2017-1-60-131
Computer Science & Engineering
East West University
Dhaka, Bangladesh
Email: 2017-1-60-131@std.ewubd.edu

Md. Nahidul Islam
ID: 2017-1-60-151
Computer Science & Engineering
East West University
Dhaka, Bangladesh
Email: 2017-1-60-151@std.ewubd.edu

Dr. Md. Golam Rabiul Alam
Associate Professor, Adjunct
Computer Science & Engineering
East West Universiy
Dhaka, Bangladesh
Email: golam.rabiul@ewubd.edu

Abstract— **Lung cancer is the uncontrolled growth of abnormal cells in one or both lungs. These abnormal cells do not carry out the functions of normal lung cells and do not develop into healthy lung tissue. This one of the leading cause of cancer deaths worldwide. A copious number of researches are going on worldwide regarding the detection of lung cancer easily by analyzing symptoms primarily before going into many complex medical tasks. Implementation of various machine learning algorithm in a medical criteria has been always a great help. In this paper, we have tried to show how much accurately various predictive models can predict presence of lung cancer from a given set of symptoms. Also, we have discussed a comparison between some classifiers which has been used for our predictive modeling.**

Keywords—Feature, Target, Predictive modeling, Naïve-Bayes, CART, KNN, SVM, Regression

I. Introduction

Medical Science is a very important area where data mining or machine learning techniques can be applied. In this present world, there are hundreds of thousands of diseases that people are suffering from. Any disease has a specific set of symptoms. Some specific combination of these symptoms causes that specific disease. So, from a number of medical checkup history of a selected disease, if a dataset can be created with the extraction of the combination of symptoms then a classifier algorithm can be trained with that dataset. Later the presence of that disease in a human body can be predicted easily only by taking the symptoms as input. In this case we have worked with lung cancer detection.

II. Problem Statement

From a machine learning point of view, we need to work on a huge data for the prediction of lung cancer presence. Research in medical science using machine learning is evolving day by day to know more about many diseases and inventing new ways of predicting the presence of them in a body. We can clearly see the recent advancement in medical technologies, which have introduced several new interesting aspects to the area. So, this is an exciting research area where we can emerge the machine learning technique for detection of many diseases. Mistakes are likely to be made in most of the cases where traditional analysis of data predicts if someone has lung cancer or not, without any complex medical exertion. The main goal of applying machine learning in this criteria is to computerize this type of analysis and improve the accuracy of the prediction. Also, development of predictive models can pave the way to use this aspect in other areas of medical research.

A. About Dataset

The given dataset was titled as lung-cancer detection. This is a Comma Separated Values (.csv) file which includes 32 rows and 57 attributes. Each row is medical records of detection of lung cancer. The first attribute denotes the result. This is the class attribute. Other 56 attributes can be denoted as many symptoms. This data was used by Hong and Young to illustrate the power of the optimal discriminant plane in ill-posed settings. In the first column data described 3 types of pathological lung cancers which are denoted with respectively 1, 2 and 3. The Authors give no information on the individual variables nor on where the data was originally used. There were some missing or irrelevant values in original dataset. All of them are replaced with a (?).

We have used five classifier algorithms to build up predictive models for lung cancer detection according to the dataset. The used classifiers are:

- Decision Tree
- Naïve Bayes
- K-Nearest Neighbor
- Support Vector Machine
- Logistic Regression

We have calculated the accuracy of each model by five time cross validation. Also F1 scores have been calculated for each model. Each models has been run on the dataset for Ten times in order to find difference between them and distinguish which one is best

III. Methodology

A. Modeling Process

For building up a predictive model, we needed to find a data set which could train our program. Then we built our program and trained with that data. We divided the collected data into train data and test data to test our program. Here's the flow chart of the process:

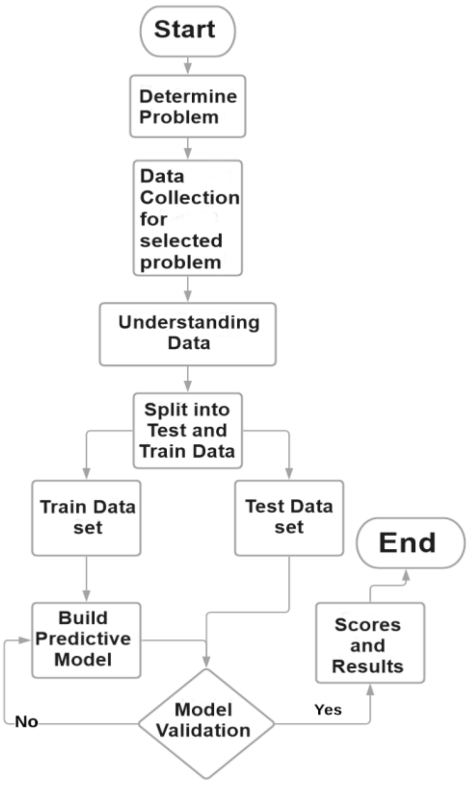


Figure 1: Flow Chart of work process

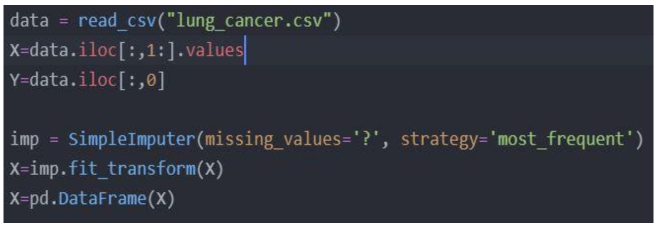
B. *Tools*

- We have used 'pandas' to read data from dataset.
- 'Numpy' has been used to handle the data and work with missing values.
- 'Scikit Learn' has been used for initiating the classifiers and train and test them with the data.
- From OS, system has been imported to draw the decision tree.
- 'Scikit Learn' has been also used to determine accuracy, f1 scores of the classifiers after testing them with the data.

C. *Using Imputer to Replace Missing Values*

Imputing means replacing missing values in a dataset by special calculations. It is already mentioned that there are 32 instances and 57 attributes in the dataset. Among them some values are denoted with the question mark (?). According to the data source, these values are actually interpreted as irrelevant in the original dataset according to the data source. Later these values were replaced with question mark (?). If we import read_csv from csv in python, it can read all the value but the classifiers cannot work with the question mark denoted data. We need imputers to replace them with numerical values.

There are various imputers in 'sklearn' interpreter. KNN Imputer, Simple Imputer & Iterative Imputer are widely used among them. In our predictive models, we have used Simple Imputer. We have set the Simple Imputer to find the most frequent value in an attribute and replace the missing values with that value. We could simply set the finder with the function of "mean" but for that the missing values should be denoted by "NaN". The figure below shows how we have replaced the missing values using imputer:



```
data = read_csv("lung_cancer.csv")
X=data.iloc[:,1:].values
Y=data.iloc[:,0]

imp = SimpleImputer(missing_values='?', strategy='most_frequent')
X=imp.fit_transform(X)
X=pd.DataFrame(X)
```

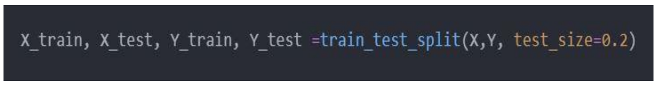
Figure 2: Code of Simple Imputing

D. *Spliting Data for Training and Testing.*

We needed to split the data for training and testing. For a good training of a classifier, huge amount of record is necessary. However, there are only 32 instances in the provided dataset.

We have taken 80% of the instances in order to train the classifiers and 20% of the instances to test the classifiers. With using this splitting we have been able to achieve good prediction by the models.

Data has been divided in to two sets, X & Y. Class Label means the Y dataset which is the first attribute in the provided dataset. Rest of the attribute has been taken into X. Then X and Y both are split into X_train, X_test and Y_train, Y_test. After that they were passed into classifiers with declaring the test size, for instance:



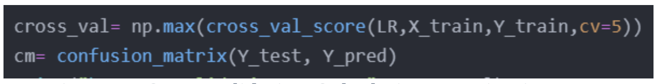
```
X_train, X_test, Y_train, Y_test =train_test_split(X,Y, test_size=0.2)
```

Figure 3: 20% of instances has been taken into consideration for testing purpose

IV. Applying Classification algorithms & Their Results

In order to compare the accuracy of various classifier to build a predictive model, we have used the same dataset to train and test different classifiers. All tested classifiers do not show the same result. They show different accuracy and f1 score. Also, difference in Confusion Matrix.

It must be noted that, each classifier algorithm has been applied on the dataset for 10 times and for each time, the cross validation score calculation criteria is set to 5.

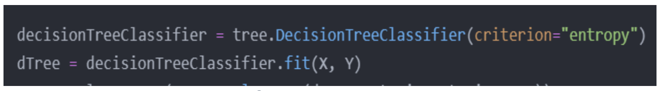


```
cross_val= np.max(cross_val_score(LR,X_train,Y_train,cv=5))
cm= confusion_matrix(Y_test, Y_pred)
```

Figure 4: Cross Validation Calculation Iteration is set to 5

A. *Applying Decision Tree Classifier Algorithm*

While training decision tree classifier algorithm we set the criterion to entropy. For that, the feature selection gets done by calculating impurity of each feature, which is the most widely used feature selection method for CART algorithm. Higher the impurity, higher chance to take the attribute in calculation.



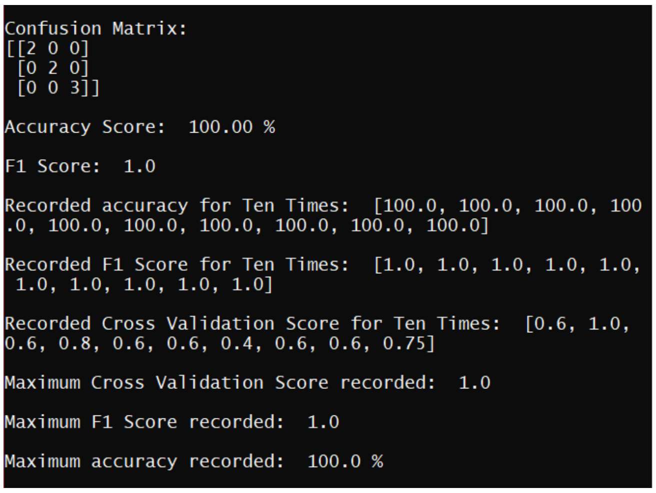
```
decisionTreeClassifier = tree.DecisionTreeClassifier(criterion="entropy")
dTree = decisionTreeClassifier.fit(X, Y)
```

Figure 5: Classifier Criterion is set to "Entropy"

We have applied the algorithm on the provided dataset for ten times and recorded the cross validation score, F1 Score and accuracy. It shows different cross validation score but always greater than 0.6. However, the accuracy is always 100%

For each calculation the confusion matrix is calculated as a diagonal matrix. Diagonal Confusion Matrix means the prediction accuracy is 100%.

Here is a screen shot of the output after applying decision tree classifier algorithm on the dataset for 10 times:



```
Confusion Matrix:
[[2 0 0]
 [0 2 0]
 [0 0 3]]

Accuracy Score:  100.00 %

F1 Score:  1.0

Recorded accuracy for Ten Times:  [100.0, 100.0, 100.0, 100
.0, 100.0, 100.0, 100.0, 100.0, 100.0, 100.0]

Recorded F1 Score for Ten Times:  [1.0, 1.0, 1.0, 1.0, 1.0,
 1.0, 1.0, 1.0, 1.0, 1.0]

Recorded Cross Validation Score for Ten Times:  [0.6, 1.0,
0.6, 0.8, 0.6, 0.6, 0.4, 0.6, 0.6, 0.75]

Maximum Cross Validation Score recorded:  1.0

Maximum F1 Score recorded:  1.0

Maximum accuracy recorded:  100.0 %
```

Figure 6: Decision Tree Output

Maximum Cross Validation Score: 1.0
Maximum F1 Score: 1.0
Maximum Accuracy: 100%
Confusion Matrix: Always Diagonal

The Decision Tree:

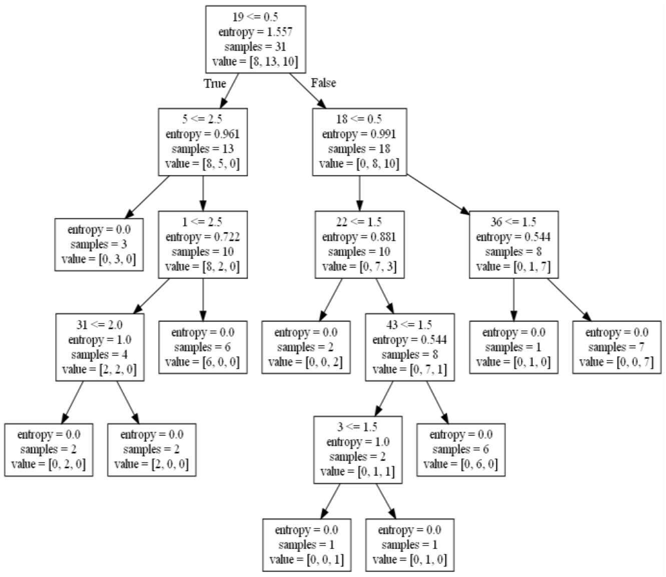


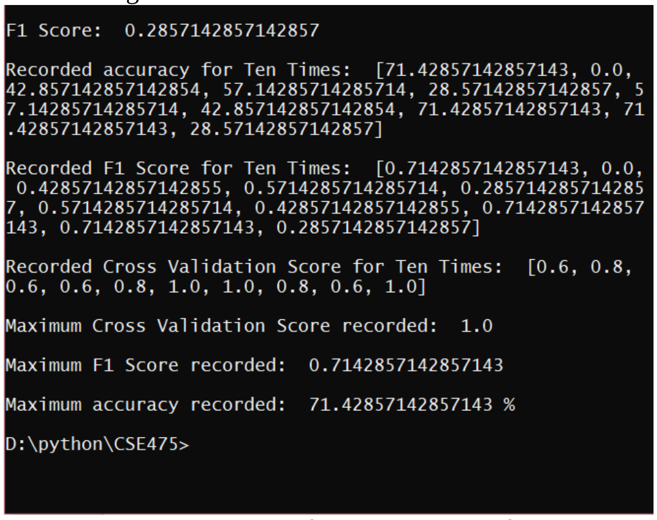
Figure 7:Generated Decision Tree

B. *Applying Naïve Bayes Algorithm*

We have used Gaussian distribution for applying Naïve Bayes classifier algorithm on the provided dataset.

Like decision tree algorithm we also ran this classifier for ten times and recorded the cross validation score, F1 Score and accuracy. It shows various accuracies, F1 Scores. But the cross validation score is always greater than 0.6. Also, the confusion matrix is not always diagonal.

Here is a screen shot of the output after applying Naïve Bayes classifier algorithm on the dataset for 10 times:



```
F1 Score:  0.2857142857142857

Recorded accuracy for Ten Times:  [71.42857142857143, 0.0,
42.857142857142854, 57.14285714285714, 28.57142857142857, 5
7.14285714285714, 42.857142857142854, 71.42857142857143, 71
.42857142857143, 28.57142857142857]

Recorded F1 Score for Ten Times:  [0.7142857142857143, 0.0,
 0.42857142857142855, 0.5714285714285714, 0.285714285714285
7, 0.5714285714285714, 0.42857142857142855, 0.7142857142857
143, 0.7142857142857143, 0.2857142857142857]

Recorded Cross Validation Score for Ten Times:  [0.6, 0.8,
0.6, 0.6, 0.8, 1.0, 1.0, 0.8, 0.6, 1.0]

Maximum Cross Validation Score recorded:  1.0

Maximum F1 Score recorded:  0.7142857142857143

Maximum accuracy recorded:  71.42857142857143 %

D:\python\CSE475>
```

Figure 8: Output of Naive Bayes Classifier

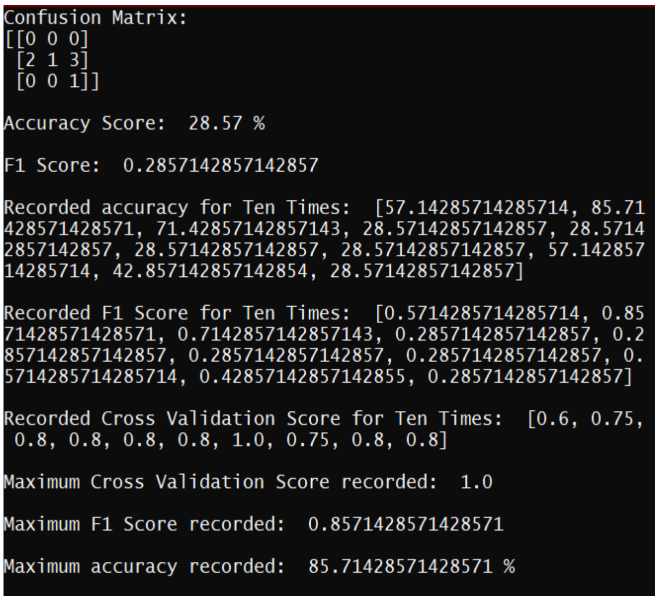
Maximum Cross Validation Score: 1.0
Maximum F1 Score: 0.7142
Maximum Accuracy: 71.42%
Confusion Matrix: Not Always Diagonal

C. *Applying Logistic Regression*

We also wanted to show how well a linear model can perform. So we have applied logistic regression algorithm on the provided dataset.

Also for this algorithm, we applied this on our dataset for 10 times. Like Naïve Bayes algorithm, the confusion matrix is not always diagonal for Logistic Regression. F1 score and accuracy vary a lot. Cross Validation Score is always greater than 0.6.

Screenshot of the output after applying Logistic Regression Algorithm has been given in next page:



```
Confusion Matrix:
[[0 0 0]
 [2 1 3]
 [0 0 1]]

Accuracy Score:  28.57 %

F1 Score:  0.2857142857142857

Recorded accuracy for Ten Times:  [57.14285714285714, 85.71
428571428571, 71.42857142857143, 28.57142857142857, 28.5714
2857142857, 28.57142857142857, 28.57142857142857, 57.142857
14285714, 42.857142857142854, 28.57142857142857]

Recorded F1 Score for Ten Times:  [0.5714285714285714, 0.85
71428571428571, 0.7142857142857143, 0.2857142857142857, 0.2
857142857142857, 0.2857142857142857, 0.2857142857142857, 0.
5714285714285714, 0.42857142857142855, 0.2857142857142857]

Recorded Cross Validation Score for Ten Times:  [0.6, 0.75,
 0.8, 0.8, 0.8, 0.8, 1.0, 0.75, 0.8, 0.8]

Maximum Cross Validation Score recorded:  1.0

Maximum F1 Score recorded:  0.8571428571428571

Maximum accuracy recorded:  85.71428571428571 %
```

Figure 9: Logistic Regression Output

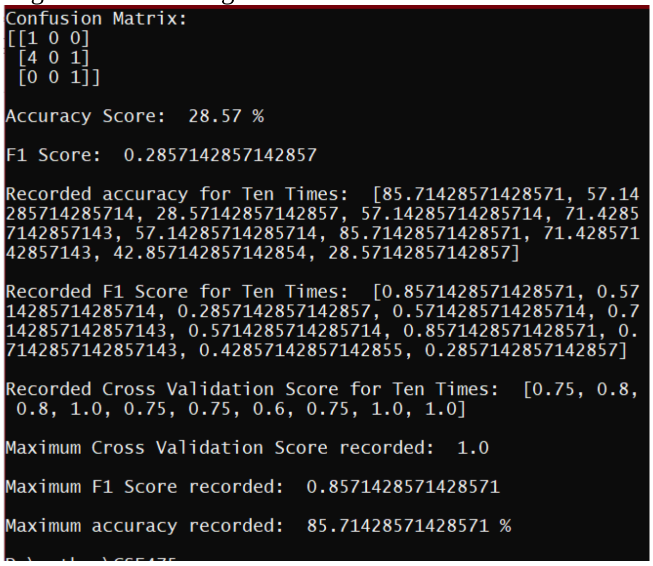
Maximum Cross Validation Score: 1.0
Maximum F1 Score: 0.8571
Maximum Accuracy: 85.71%
Confusion Matrix: Not Always Diagonal

D. Applying K- Nearest Neighbor Algorithm

K-Nearest Neighbor is the simplest supervised learning algorithm. Applying the K-Nearest Neighbor Algorithm, we found out that the result is very much similar to Logistic regression.

Like all other algorithms, we have applied this algorithm on our dataset for 10 times. The result being so much similar to Logistic Regression result, the confusion matrix is not always diagonal for K-Nearest Neighbor. Also, F1 score and accuracy vary a lot. Cross Validation Score is always greater than 0.6.

Screenshot of the output after applying Logistic Regression Algorithm has been given below:



```
Confusion Matrix:
[[1 0 0]
 [4 0 1]
 [0 0 1]]

Accuracy Score:  28.57 %

F1 Score:  0.2857142857142857

Recorded accuracy for Ten Times:  [85.71428571428571, 57.14
285714285714, 28.57142857142857, 57.14285714285714, 71.4285
7142857143, 57.14285714285714, 85.71428571428571, 71.428571
42857143, 42.857142857142854, 28.57142857142857]

Recorded F1 Score for Ten Times:  [0.8571428571428571, 0.57
14285714285714, 0.2857142857142857, 0.5714285714285714, 0.7
142857142857143, 0.5714285714285714, 0.8571428571428571, 0.
7142857142857143, 0.42857142857142855, 0.2857142857142857]

Recorded Cross Validation Score for Ten Times:  [0.75, 0.8,
 0.8, 1.0, 0.75, 0.75, 0.6, 0.75, 1.0, 1.0]

Maximum Cross Validation Score recorded:  1.0

Maximum F1 Score recorded:  0.8571428571428571

Maximum accuracy recorded:  85.71428571428571 %
```

Figure 10: K-th Nearest Neighbor Output

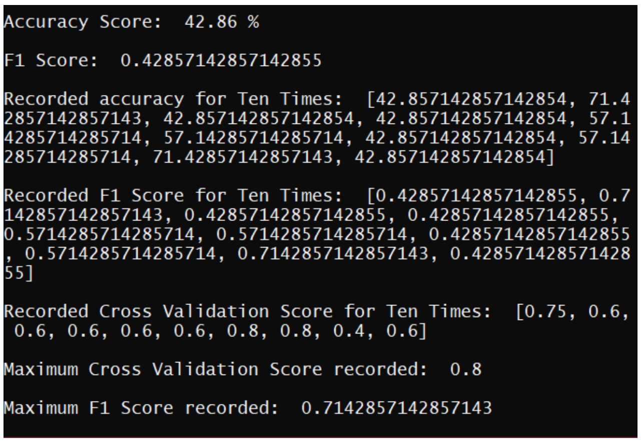
Maximum Cross Validation Score: 1.0
Maximum F1 Score: 0.8571
Maximum Accuracy: 85.71%
Confusion Matrix: Not Always Diagonal

E. Applying Support Vector Classifier Algorithm

Using Support Vector as classifier is also one of the simplest supervised learning algorithm. Applying this algorithm we achieved a result which is similar to the result given by naïve bayes algorithm.

This algorithm was also ran for 10 time on the dataset. The confusion matrix is sometimes diagonal, F1 score and accuracy vary a lot and the cross validation score is always greater than 0.6 but maximum is 0.8.

Here is a screenshot of the output after applying support vector classifier algorithm:



```
Accuracy Score:  42.86 %

F1 Score:  0.42857142857142855

Recorded accuracy for Ten Times:  [42.857142857142854, 71.4
2857142857143, 42.857142857142854, 42.857142857142854, 57.1
4285714285714, 57.14285714285714, 42.857142857142854, 57.14
285714285714, 71.42857142857143, 42.857142857142854]

Recorded F1 Score for Ten Times:  [0.42857142857142855, 0.7
142857142857143, 0.42857142857142855, 0.42857142857142855,
0.5714285714285714, 0.5714285714285714, 0.42857142857142855
, 0.5714285714285714, 0.7142857142857143, 0.428571428571428
55]

Recorded Cross Validation Score for Ten Times:  [0.75, 0.6,
 0.6, 0.6, 0.6, 0.6, 0.8, 0.8, 0.4, 0.6]

Maximum Cross Validation Score recorded:  0.8

Maximum F1 Score recorded:  0.7142857142857143
```

Figure 11: Support Vector Classifier Output

Maximum Cross Validation Score: 0.8
Maximum F1 Score: 0.7142
Maximum Accuracy: 71.42%
Confusion Matrix: Not Always Diagonal

V. DISCUSSION

From the results section if we collect and show the average results in a table:

Classifier Algorithm	**Confusion Matrix**	**Max F1 Score**	**Max Accuracy**	**Max Cross validation Score**
Decision tree	Always Diagonal	1.0	100%	1.0
Naïve bayes	Not always diagonal	0.714	71.42%	1.0
Logistic Regression	Not always diagonal	0.857	85.71%	1.0
Kth-Nearest Neighbor	Not always diagonal	0.714	71.42%	1.0
Support Vector Classifier	Not always diagonal	0.714	71.43%	0.8

We were instructed to use more than one classifier algorithm on the provided dataset to extract the accuracy of each one of them in lung cancer detection.

Among all the classifiers, applying a CART algorithm, Decision Tree, has given the most accurate result. It is mainly because of the data provided, is non-parametric. Decision tree algorithm works very well on non-parametric data. Also, decision tree uses the “divide and conquer” method, it tends to perform well if there are a lot of highly relevant attributes. So, it is being able to distinguish the importance of attributes and eliminating the attributes which is not important for prediction calculation. No other classifiers is doing that for this dataset. It must be noted that there are 56 attributes in the dataset and it must be distinguished that which attributes shows more importance and are likely to be selected for calculation.

The accuracy 84.3% provides us the assurance that the program we have built is reliable. Also, if the program is trained with more music data, for instance, 10 or 15 thousand of data, the program’s prediction will be more accurate.

One disadvantage of our program is, our code is not generalized. If a new feature is needed to be added in the train or test data, then some changes must be made in the code.

In our train data, we noticed a complex interactions between the features. However, Naïve Bayesian classifiers and logistic regression classifier assume that there are no dependencies amongst attributes. This assumption is called class conditional independence. It is made to simplify the computations involved and, hence is called "naive". But actually the attributes are a little dependent on each other (not very much). That is why the naïve Bayes classifier and logistic regression are failing to show stable accuracy. Sometimes the accuracy is high and sometimes it is low.

On the other hand, Kth Nearest Neighbor (KNN) algorithm and Support Vector Classifier (SVC) algorithm are very simple and supervised learning algorithm. They do not tend to work good on complexly interactive attributes. This is why, like naïve bayes and logistic regression, these two algorithms are also showing various results. Also, it must be taken into consideration that the cross validation score of KNN is 1.0 where cross validation score of SVC is 0.8. So, KNN is being a better approach than SVC in this criteria. Which However, for simple and fine datasets, these algorithms are very supportive.

CONCLUSION

We have trained and tested many classifier algorithm with same dataset and tried to predict the presence of lung cancer taking 56 different symptoms in consideration. The reason behind using more than one algorithm is to show the difference between them and find out which is best for creating a predictive model for lung cancer detection. This kind of research plays a very important role in the world of medical sciences. So, more and more involvement of machine learning and data mining techniques in similar aspects is very necessary. Concluding, we realized that this project is only a baby step towards the world of data science. This project has made us more interested to work with big data and we are eagerly waiting to contribute in more complex and bigger projects in future.

ACKNOWLEDGMENT

The project has been done with the optimum contribution of three group members, *Abdullah Muhammad Tahir, Tanvir Ahmed* and *Md. Nahidul Islam,* students of Spring 2017, *East West University, Dhaka.*

This enormous thanks to our course instructor, *Prof. Dr. Md. Golam Rabiul Alam,* Associate Professor (Adjunct), *Department of Computer Science & Engineering, East West University,* Dhaka. With his expert supervision and kind help, we have completed this project.