CHAPTER 5
IMPLEMENTATION AND RESULTS

5.1. Implementation

The implementation chapter is a chapter on the narrative of the utilization of data structure and algorithms in the form of applied applications. Starting from pre-processing the dataset so that it can be processed in the Naïve Bayes classifier which is then tested to produce values for the *confusion matrix, accuracy, precision, recall,* and *f1 score.*

The Wisconsin dataset used in this study was obtained from the UCI Repository. After the dataset is converted, it can then be entered into the classifier.

This sub-chapter contains the input process for the Wisconsin breast cancer dataset which has been converted to data by deleting 16 string-type data which is notated with "?" to simplify the data processing process. So the amount of data becomes less than the raw data.

The programming language used to classify the Wisconsin breast cancer dataset in this study is the Python programming language that uses a library called Scikit-learn. Sklearn has an efficient data structure making it easy to use for data mining and data analysis. This library contains several machine learning algorithms such as classification, clustering and pre processing. The tool used is Jupyter Notebook for the Python programming language.

For the first step, we need to import and read dataset. The dataset saved in BCData.csv

```
1. breast_cancer=pd.read_csv('C:\\Users\Hp\Downloads\jurnal yg
   dipakai\BCData.csv', delimiter=';')
2. breast_cancer
3. data = breast_cancer.drop('Sample_code_number', axis=1)
```

Line 1 is the input process and reads the *BCData.csv* file for the classification process. The 2nd line is for displaying the dataset. The Wisconsin dataset has 10 attributes and 1 class attribute. In this classification process, all attributes will be used in addition to the sample_code_number. So in line 3 , the `drop()` function is carried out for removing the sample_code_number attribute label, then the new dataset is stored with the name variable `data`.

```
4. data1 = data.loc[~data.eq('?').any(1)]
```

```
5. X=data1.drop('Class',axis=1)
6. y= data1.Class
```

In line 4 to find and delete the missing value in the form of "?". Lines 5 and 6 are functions to determine the independent and dependent variables.

```
7. X_train, X_test, y_train, y_test= train_test_split(X,y,test_size=0.3,
   shuffle=True, random_state=0)
8. print("shape of original dataset :", df1.shape)
9. print("\nshape of input data training :", X_train.shape)
10.   print ("\nshape of input data testing :", X_test.shape)
11.   prioryy2 = len(Xy2) / len(X_train)
12.   priory4 = len(Xy4) / len(X_train)
13.   print(priory2, priory4)
```

We can see in Line 7, it aim to split data set into train as X_train for all attributes except Class attributes and test data with a ratio of 70:30. Here the value for **`random_state=0`**, it means that every time line 7 is run a new random value is generated and the train test data set will have a different value each time. Line 8, 9, and 10 just to print the shape of each. Function `prior()` in Line 11,12 is to calculate the prior probability of each class.

```
14.   model = GaussianNB()
15.   nbTrain= model.fit(X_train, y_train)
```

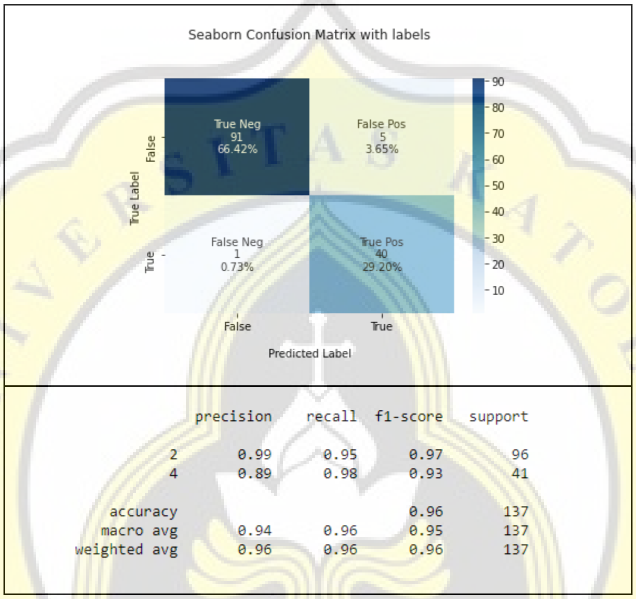
`GaussianNB()` is a function to implement the Gaussian Naïve Bayes algorithm. The `fit()` function is used to train the model. Then to measure accuracy value in Naïve Bayes, `confusion_matrix()` function in Line 16 is needed.

```
16.   cm = confusion_matrix(arr, y_prediction)
17.   from sklearn.metrics import classification_report
18.   print(classification_report(arr, y_prediction))
```

The results are interpreted into a 2x2 matrix as True Negative, False Positif, False Negative, and True Positif. From the results of the confusion matrix can also be calculated for the value of accuracy, precision, recall, and F1 score using the `classification_report()` function from `sklearn.metrics` in Line 17

5.2. Testing

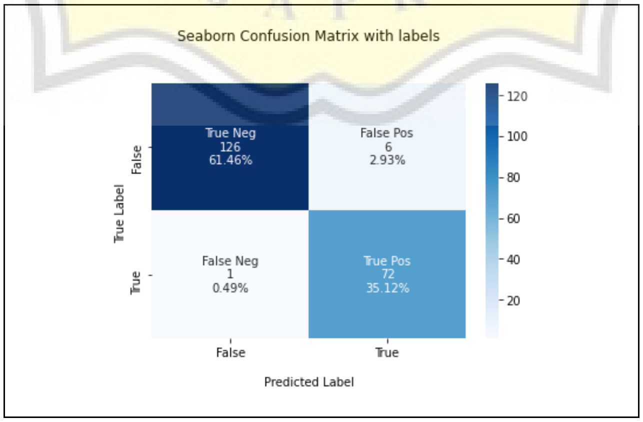
This research has been carried out using 683 breast cancer datasets taken based on the values of Clump Thickness, Uniformity of Cell Size, Uniformity of Cell Shape, Marginal Adhesion, Single Epithelial Cell Size, Bare Nuclei, Bland Chromatin, Normal Nucleoli, Mitoses and classified into benign and malignant classes. At the testing using Naïve Bayes with a training and testing data ratio of 80:20, the results obtained were 95.62% of the data that could be classified correctly and 4.38% of the data that could not be classified correctly.

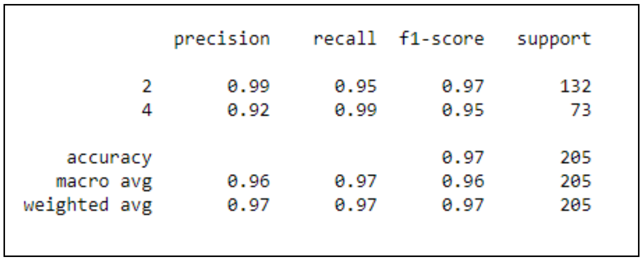


```
              precision    recall  f1-score   support

           2       0.99      0.95      0.97        96
           4       0.89      0.98      0.93        41

    accuracy                           0.96       137
   macro avg       0.94      0.96      0.95       137
weighted avg       0.96      0.96      0.96       137
```

Figure 5.1 Confusion Matrix 80:20

The next step is to do a calculation with a training and testing data ratio of 70:30, the results obtained are 96.58% of the data that can be classified correctly and 3.42% cannot be classified correctly.

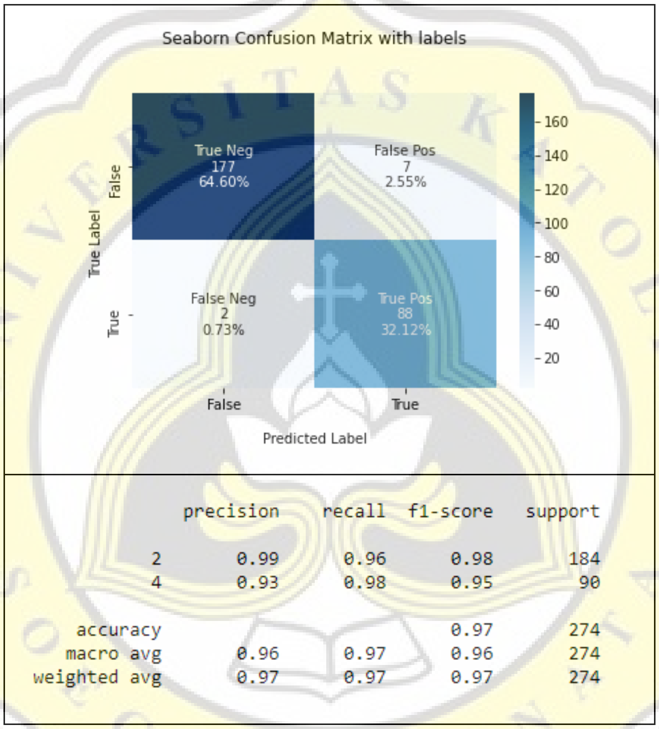




```
              precision    recall  f1-score   support

           2       0.99      0.95      0.97       132
           4       0.92      0.99      0.95        73

    accuracy                           0.97       205
   macro avg       0.96      0.97      0.96       205
weighted avg       0.97      0.97      0.97       205
```

Figure 5.2 Confusion Matrix 70:30

In the trial using a ratio of 60:40, it was found that 96.72% of the data were classified correctly, and 3.28% of the data could not be classified correctly.



```
              precision    recall  f1-score   support

           2       0.99      0.96      0.98       184
           4       0.93      0.98      0.95        90

    accuracy                           0.97       274
   macro avg       0.96      0.97      0.96       274
weighted avg       0.97      0.97      0.97       274
```

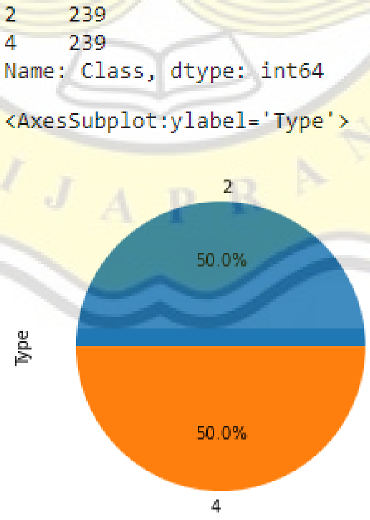
Figure 5.3 Confusion Matrix 60:40

From several experiments it can be seen that the predicted value of True Positive is smaller than the value of True Negative. This is appropriate because the dataset used has a higher probability value for the Benign class.

Table 5.1. Test Results

Try to -	Percentage Split	Data Training	Data Testing	Percentage of Correctly Classified Data	Percentage of Data Classified Wrong	Precision	Recall
1	95%	34	649	95,84%	4,16%	0,95	0,965
2	90%	69	614	98,55%	1,45%	0,985	0,985
3	80%	136	547	96,34%	3,66%	0,955	0,965
4	70%	204	479	96,25%	3,76%	0,955	0,965
5	60%	273	410	96,34%	3,66%	0,955	0,965
6	50%	341	342	96,49%	3,50%	0,955	0,965
7	40%	409	274	96,72%	3,28%	0,96	0,97
8	30%	478	205	96.58%	3.42%	0,955	0,97
9	20%	546	137	95,62%	4,38%	0,94	0,965
10	10%	614	69	98,55%	1,45%	0,985	0,985

Table 5.1 presents the test results of the percentage of data classified as true or false, so in this study precision and recall were also used to measure the performance of the application of Naïve Bayes to the prediction of breast cancer. It can be said that precision measures the quality of classification while recall measures the quantity of classification. In this study, the positive data is data on the class of benign. While the negative data is data on the class of malignant.

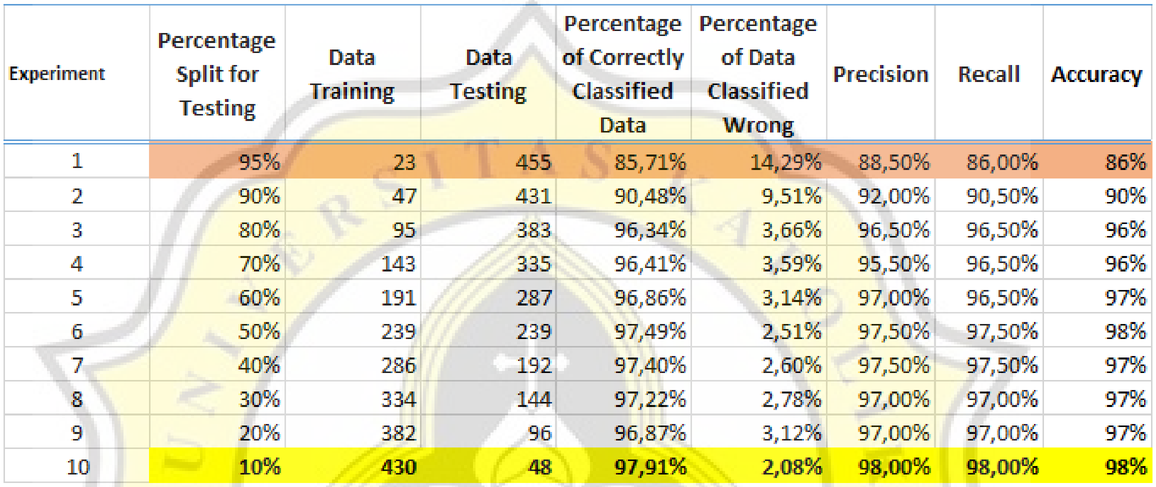


```
2    239
4    239
Name: Class, dtype: int64

<AxesSubplot:ylabel='Type'>
```

Figure 5.4 Balanced Data Down

For the last test, we tried to use balanced data where the number of benign and malignant classes was the same. Previously, the number for each benign class was 444 and the malignant class was 239. The downsampling technique was used here by reducing the number of data in the majority class. Can be seen in the Figure 5.4, so that at the end the number for each class is the same, which is 239.

Table 5.2. Test Result for Balanced Data



Experiment	Percentage Split for Testing	Data Training	Data Testing	Percentage of Correctly Classified Data	Percentage of Data Classified Wrong	Precision	Recall	Accuracy
1	95%	23	455	85,71%	14,29%	88,50%	86,00%	86%
2	90%	47	431	90,48%	9,51%	92,00%	90,50%	90%
3	80%	95	383	96,34%	3,66%	96,50%	96,50%	96%
4	70%	143	335	96,41%	3,59%	95,50%	96,50%	96%
5	60%	191	287	96,86%	3,14%	97,00%	96,50%	97%
6	50%	239	239	97,49%	2,51%	97,50%	97,50%	98%
7	40%	286	192	97,40%	2,60%	97,50%	97,50%	97%
8	30%	334	144	97,22%	2,78%	97,00%	97,00%	97%
9	20%	382	96	96,87%	3,12%	97,00%	97,00%	97%
10	**10%**	**430**	**48**	**97,91%**	**2,08%**	**98,00%**	**98,00%**	**98%**

In table 5.2 it can be seen that the highest accuracy results obtained are 98% and the lowest is 86%. From experiment 1, the value for wrong data classification is 14.29%, this shows that the amount of training data can also affect the algorithm.