CHAPTER 5

IMPLEMENTATION AND TESTING

5.1 Implementation

5.2 Vector Space Model

In this section the vector space model algorithm uses the data train that has been provided to test the test data. So that it will produce a sentiment from the test data to be tested and the results of the weight. Start by calculating tf-idf to get the weight of each letter.

```
1. for test in tfidf_test:
2.     hasilakar_test_train = []
3.     y = 0
4.     for train in hasil_train:
5.         tmp = 0
6.         for t in kataunik:
7.             tmp += train[t]*test[t]
8.         res = tmp/hasilkuadrat_test[i]*hasilkuadrat_train[y]
9.         hasilakar_test_train.append(res)
10.            y+=1
11.        hasil.append(hasilakar_test_train)
```

The code above is a cosine similarity that works in Vector Space Model. On line 7 the tmp variable will be filled with the multiplication of the training data (train) weight and the testing data (test) weight. Which then on line 8 will be divided from the square root of the weight of the training data and testing data that has been previously calculated. The results of these calculations will be calculated how close it is to the training data document and generate sentiment depending on the calculation.

5.3 Naive Bayes

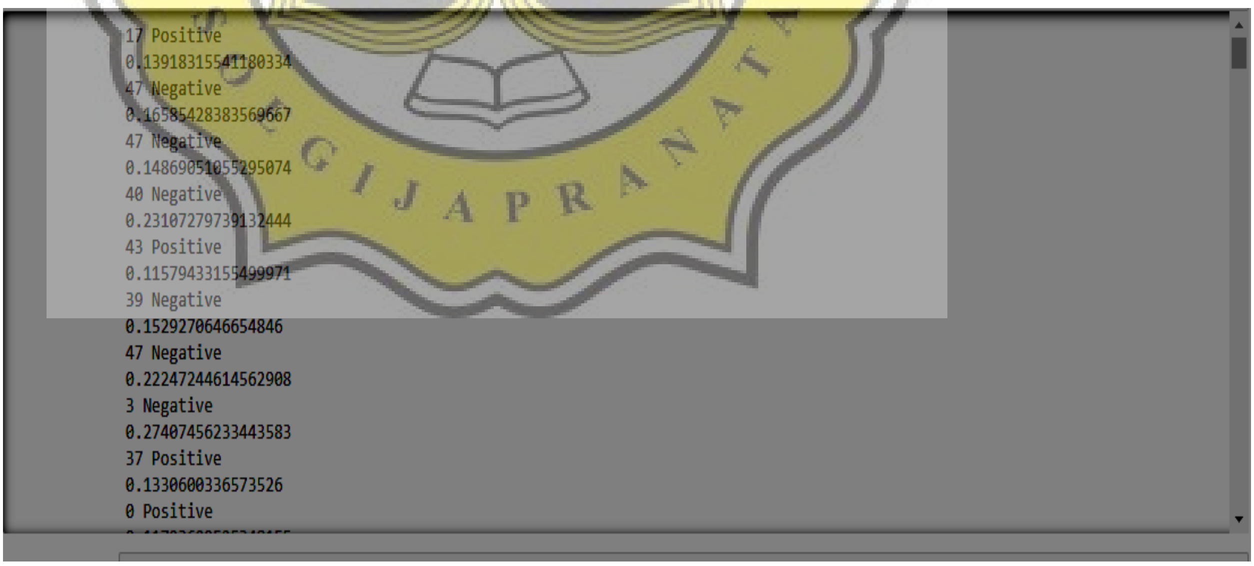
Similar to the vsm calculation, Naive Bayes also calculates the weight of each word using tf-idf. Which is then followed by using the Naive Bayes algorithm to get the sentiment results from the test data.

```
12.    from sklearn import naive_bayes
```

```
13.    Stopwords = set(stopwords.words('english'))
14.    vectorizer = TfidfVectorizer(use_idf=True, lowercase=True,
  strip_accents='ascii', stop_words=Stopwords)
15.    y=data.sentiment
16.    x=vectorizer.fit_transform(data.text)
17.    x_train,x_test,y_train,y_test=train_test_split(x,y,random_st
  ate=12)
18.    clf= naive_bayes.MultinomialNB()
19.    clf.fit(x_train,y_train)
20.    word=pd.read_csv(r"C:\Users\ASUS\Documents\skripsiveda\data\
  150test.csv")
21.    testing = word.iloc[:,0]
22.    test = vectorizer.transform(testing)
23.    test1 = clf.predict(test)
24.    print(test1)
```

The code above is a calculation for the Naive Bayes algorithm. To determine the sentiment from the test data, it is necessary to calculate the tf-idf from the training data that has positive and negative labels. Code 13 and 14 are codes for calculating tf-idf using the library and set stopwords to English. Then on line 18 is a library from Naive Bayes that uses multinomials.

5.4 Testing

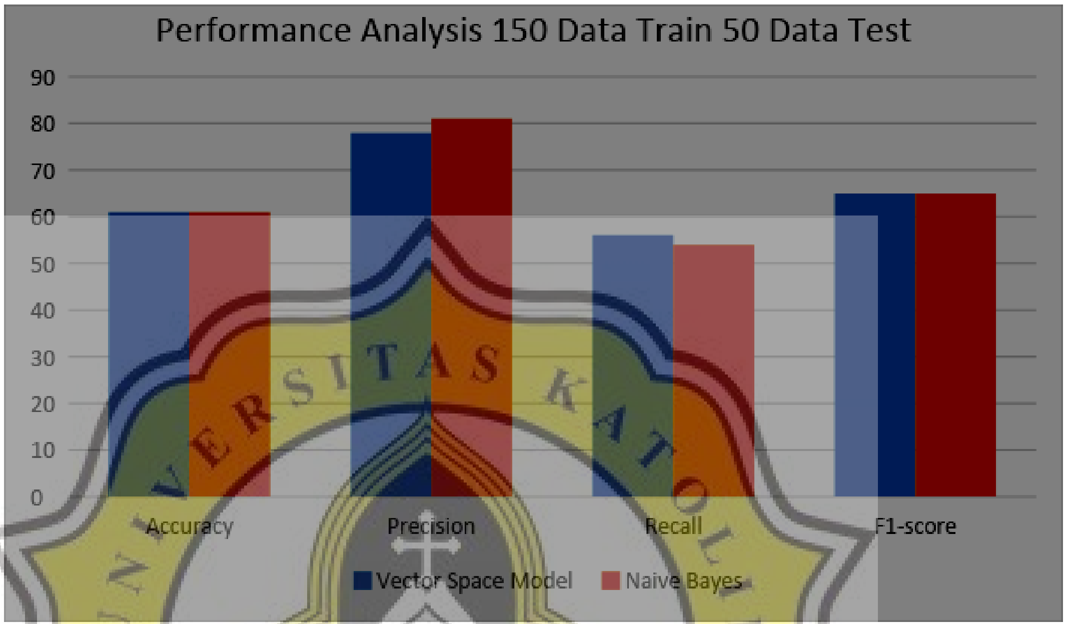


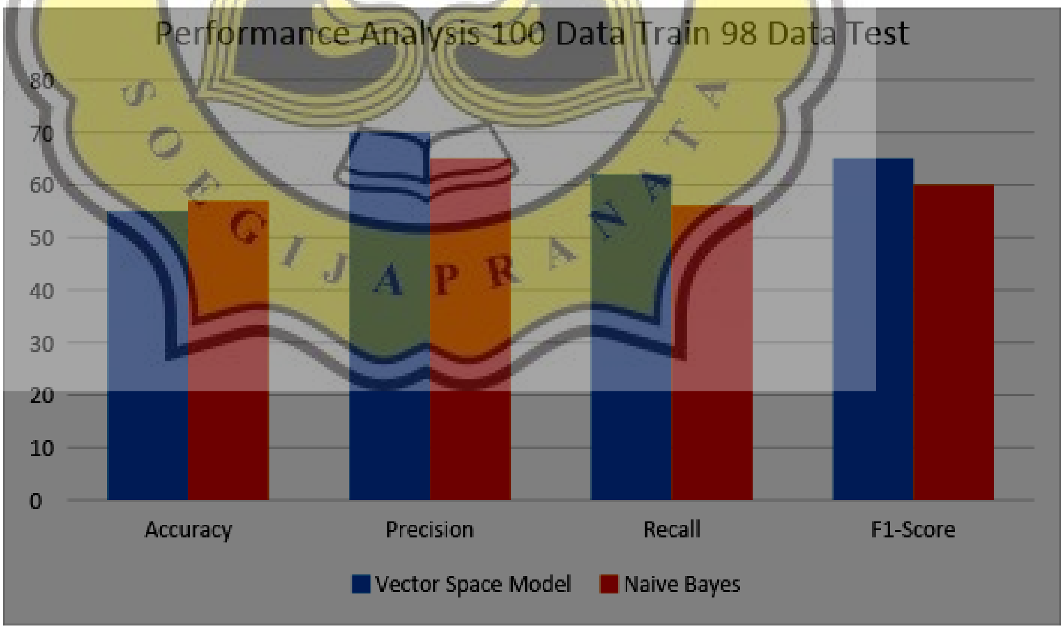
In the VSM method, there are three results obtained, namely a training data document that is similar to testing data, then the sentiment obtained from the algorithm's prediction and the weight of the test data obtained from the above calculations. Which later will be obtained the results of TP, TN, FP, FN by manually checking. Then calculated to get accuracy, precision, recall and f1-score.

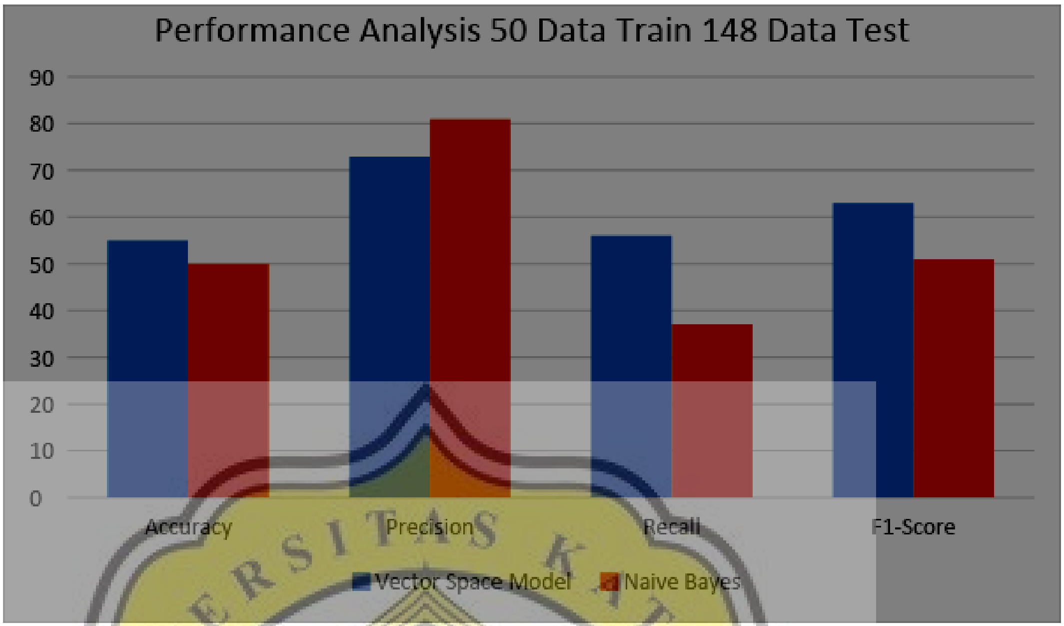
```
['Negative' 'Negative' 'Positive' 'Negative' 'Positive' 'Negative'
 'Negative' 'Negative' 'Negative' 'Negative' 'Positive' 'Positive'
 'Negative' 'Negative' 'Negative' 'Negative' 'Negative' 'Negative'
 'Negative' 'Negative' 'Negative' 'Positive' 'Positive' 'Negative'
 'Negative' 'Negative' 'Negative' 'Negative' 'Positive' 'Positive'
 'Negative' 'Negative' 'Negative' 'Negative' 'Positive' 'Negative'
 'Negative' 'Negative' 'Negative' 'Positive' 'Positive' 'Negative'
 'Negative' 'Negative' 'Negative' 'Negative' 'Negative' 'Negative'
 'Negative' 'Negative' 'Negative' 'Positive' 'Positive' 'Positive'
 'Negative' 'Negative' 'Positive' 'Negative' 'Negative' 'Negative'
 'Negative' 'Positive' 'Positive' 'Negative' 'Negative' 'Negative'
 'Negative' 'Negative' 'Positive' 'Positive' 'Negative' 'Negative'
 'Positive' 'Negative' 'Positive' 'Negative' 'Positive' 'Negative'
 'Negative' 'Positive' 'Negative' 'Positive' 'Negative' 'Positive'
 'Positive' 'Positive' 'Negative' 'Negative' 'Positive' 'Negative'
 'Negative' 'Negative' 'Negative' 'Negative' 'Negative' 'Negative'
 'Positive' 'Positive' 'Negative' 'Negative' 'Positive' 'Positive'
 'Negative' 'Negative' 'Negative' 'Negative' 'Negative' 'Positive'
 'Positive' 'Negative' 'Negative' 'Negative' 'Negative' 'Positive'
 'Negative' 'Negative' 'Negative' 'Negative' 'Positive' 'Negative'
 'Negative' 'Positive' 'Negative' 'Positive' 'Negative' 'Positive'
 'Positive' 'Positive' 'Negative' 'Negative' 'Positive' 'Negative'
 'Negative' 'Positive' 'Negative' 'Negative' 'Negative' 'Negative'
 'Positive' 'Positive' 'Negative' 'Negative' 'Negative' 'Negative'
 'Negative' 'Positive' 'Positive' 'Negative']
```

The following is a calculation of the Naive Bayes algorithm. The results obtained from calculations using Naive Bayes are only the sentiment results from the test data. What will be the same will also be calculated for accuracy, precision, recall and f1-score by manually checking to get TP, TN, FP, FN.

From the tests that have been carried out, the following is the final result obtained from this project.







With the results of the chart above, it is found that these two algorithms have almost the same performance. Because these two algorithms are both classified as supervised algorithms. With a maximum level of performance in the scheme of 150 training data and 50 data testing.

	A	B	C	D	E	F	G	H
1	VSM	Label	Naive Bayes		tp	18	a	61
2	69 Positive	2	Positive		tn	12	p	78
3	90 Positive	1	Negative		fp	14	r	56
4	84 Negative	1	Negative		fn	5	f	65
5	33 Negative	2	Negative		Vector Space Model			
6	34 Negative	2	Negative					
7	35 Positive	2	Negative		tp	18	a	61
8	36 Positive	1	Positive		tn	12	p	81
9	37 Negative	2	Negative		fp	4	r	54
10	38 Negative	1	Positive		fn	15	f	65
11	39 Positive	1	Negative		Naive Bayes			
12	25 Negative	1	Positive					
13	41 Positive	1	Positive					
14	42 Positive	1	Positive					
15	43 Positive	1	Positive					
16	69 Positive	1	Positive					
17	128 Positive	1	Positive					
18	104 Negative	2	Negative					
19	69 Positive	2	Positive					
20	70 Negative	1	Negative					
21	90 Positive	2	Negative					
22	69 Positive	1	Positive					
23	70 Negative	1	Negative					

Sheet1

Illustration 5.1: Evaluation

The picture above is a way to evaluate the program that has been made. By manually calculating how many TP, TN, FP, FN, the accuracy, precision, recall and F1-score are obtained. With column A for label prediction from VSM, while column B is the correct label and has been checked manually then column C is for Naive Bayes. The following is a calculation of accuracy, precision, recall and F1-Score for Vector Space Model and Naive Bayes from the image above.

Vector Space Model

$$Accuracy = \frac{(TP + TN)}{(TP + FP + FN + TN)} = \frac{(18 + 12)}{(18 + 14 + 5 + 12)} = 61\%$$

$$Presisi = \frac{(TP)}{(TP + FP)} = \frac{(18)}{(18 + 14)} = 78\%$$

$$Recall = \frac{(TP)}{(TP + FN)} = \frac{(18)}{(18 + 5)} = 56\%$$

$$F1 - Score = \frac{2 \times (Recall \times Precision)}{(Recall + Precision)} = \frac{(78 \times 56)}{(78 + 56)} = 65\%$$

Naive Bayes

$$Accuracy = \frac{(TP + TN)}{(TP + FP + FN + TN)} = \frac{(18 + 12)}{(18 + 4 + 15 + 12)} = 61\%$$

$$Presisi = \frac{(TP)}{(TP + FP)} = \frac{(18)}{(18 + 4)} = 81\%$$

$$Recall = \frac{(TP)}{(TP + FN)} = \frac{(18)}{(18 + 15)} = 54\%$$

$$F1 - Score = \frac{2 \times (Recall \times Precision)}{(Recall + Precision)} = \frac{(81 \times 54)}{(81 + 54)} = 65\%$$

.