

# **CHAPTER 4**

## **ANALYSIS AND DESIGN**

### **4.1. Analysis**

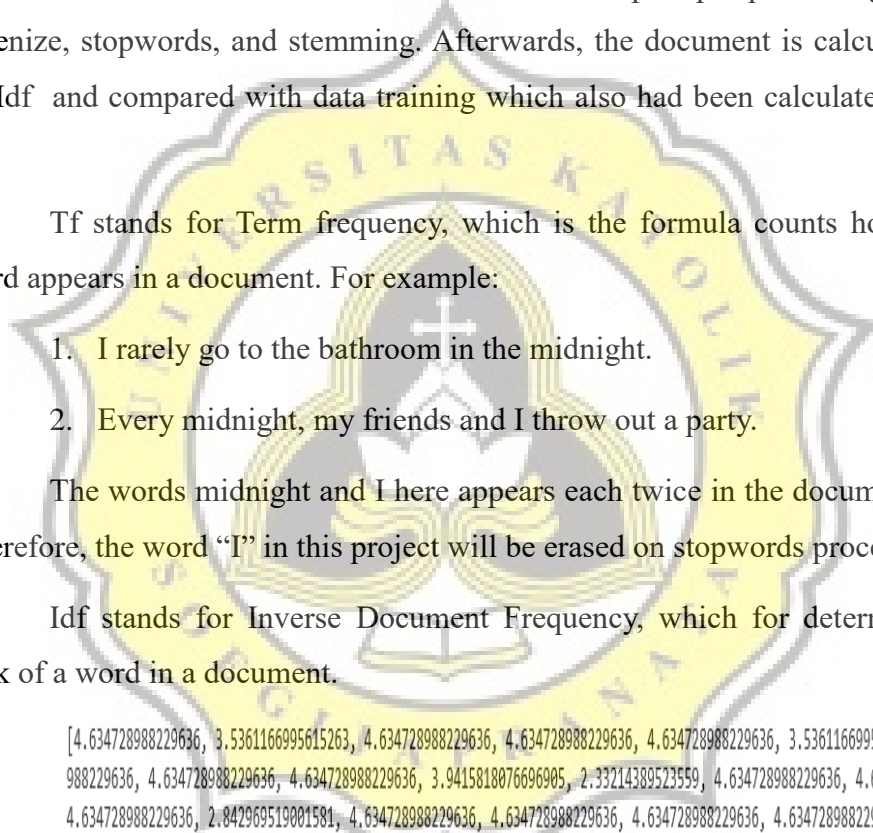
The project use python as its programming language. The first thing came with scrapping tweets by #hashtag trending from twitter, data will be saved in csv form and its title has # on the front. The next step is pre-processing which is tokenize, stopwords, and stemming. Afterwards, the document is calculated with Tf-Idf and compared with data training which also had been calculated with Tf-Idf.

Tf stands for Term frequency, which is the formula counts how often a word appears in a document. For example:

1. I rarely go to the bathroom in the midnight.
2. Every midnight, my friends and I throw out a party.

The words midnight and I here appears each twice in the document above. Therefore, the word "I" in this project will be erased on stopwords process.

Idf stands for Inverse Document Frequency, which for determining the rank of a word in a document.



```
[4.634728988229636, 3.5361166995615263, 4.634728988229636, 4.634728988229636, 4.634728988229636, 3.5361166995615263, 4.634728988229636, 4.634728988229636, 4.634728988229636, 3.9415818076696905, 2.33214389523559, 4.634728988229636, 4.634728988229636, 4.634728988229636, 2.842969519001581, 4.634728988229636, 4.634728988229636, 4.634728988229636, 4.634728988229636, 4.634728988229636, 4.634728988229636, 4.634728988229636, 4.634728988229636, 4.634728988229636, 4.634728988229636, 4.634728988229636, 4.634728988229636, 4.634728988229636, 4.634728988229636, 3.9415818076696905, 4.634728988229636, 4.634728988229636, 4.634728988229636, 4.634728988229636, 3.5361166995615263, 4.634728988229636, 4.634728988229636, 4.634728988229636, 4.634728988229636, 4.634728988229636, 4.634728988229636, 4.634728988229636, 4.634728988229636, 4.634728988229636]
```

**Illustration 4.1.1: Idf scores**

Appearance based-on score:

4.63 = only 1 appearance

3.56 = 2 or 3 appearances

2.33 = 4 or more appearances

The Final step is all the result input into k-NN algorithm to find out the nearest neighbour of negative words which is the lowest score.

$$\cos(\Theta_{ij}) = \frac{\sum_k (d_{ik} d_{jk})}{\sqrt{\sum_k d_{ik}^2} \sqrt{\sum_k d_{jk}^2}}$$

Illustration 4.1.2: Euclidian Distance Formula

$d_{ik}$  = data training

$d_{jk}$  = data testing

$\Sigma$  = Sigma (Total Point)

The results is the document standard. If the result is closed to minimal score is called negative, otherwise if the results is the same as minimal score called true negative.

Document	Value	Object	KNN
1	3.6677901234544	NEGATIVE	NEGATIVE
2	4.69639491459854	POSITIVE	POSITIVE
3	19.4731589849157	NEGATIVE	TRUE NEGATIVE

Table 4.1: k-Nearest Neighbour Analytical Table

Final conclusion of this project is the result which has less point has the nearest score on negative text. The document which declared negative is the document

which has more than 40% negative texts inside. The accuracy of this calculation is 82,3%.

## 4.2. Design

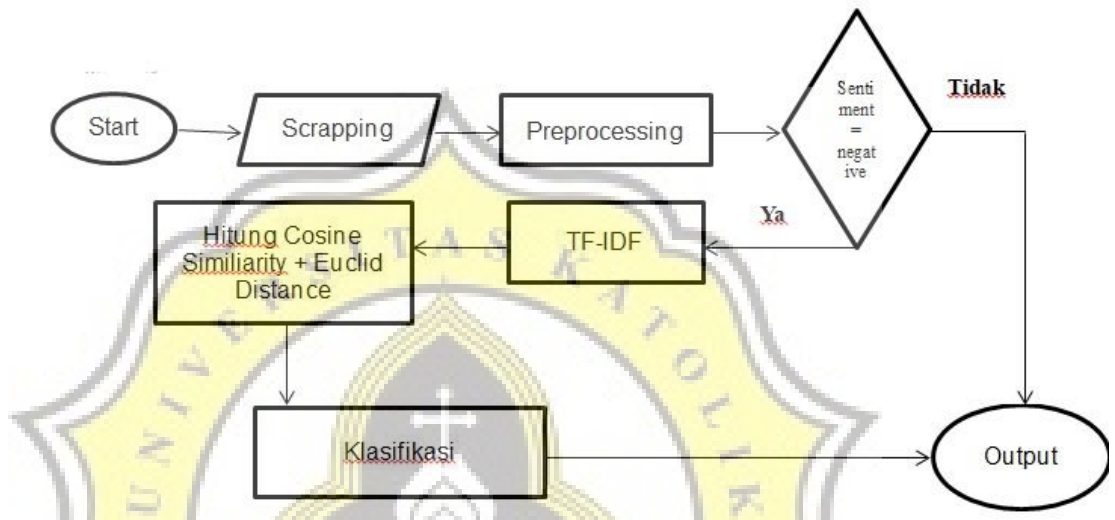


Illustration 4.2: Procedure of calculating k-Nearest Neighbour

After the scrapping step is complete, every training data contained in the document must be label before the pre-processing step begins. The next step is to calculate the Tf-Idf of each document that has been previously scraped. save the results of the Tf-Idf calculation earlier. Next is to calculate the Tf-Idf from existing training data.

The final step is the calculation of k-Nearest Neighbour, training data minus testing data and then the results are in rank two. The results of this count are divided by the total amount of data in the document. the results of the division will be squared which will later be a score for determining the type of data.