**CSI4107 Assignment 2: Sentiment Analysis in Twitter Messages**

**Classifier Descriptions**

For this assignment, decided to run a number of different classifiers to run our dataset through. The classifiers had the responsibility of predicting if a tweet was positive, negative, neutral or objective. The following classifiers were the chosen algorithms used to determine the category of the tweets.

A decision tree (J48)

This classifier will iteratively match with the best attribute to split on by using the method of gain ratio in order to overcome the bias to multi-valued attributes that tend to happen with information gain. The algorithm recurses and essentially splits the subsets by the next attribute with the greatest information gain to produce the following subsets. This algorithm also handles pruning the tree to limit overfitting, this can be illustrated with words such as “bad” which would be a good split point for determining a classification which would then decide whether or not to

branch further.

K-Nearest neighbor (IBk)

The K-Nearest neighbor classifier stores all available cases and classifies new cases using

a distance function such as Euclidean, Hamming, Minkowski or Manhattan. This classifier is

specified as lazy learning approach since it spends more time during testing than it does during

training. It works by computing the k closest neighbor to each testing instance and assigns it to the class with the highest number of the closest neighbor. This classifier also holds the property of having k as an odd integer to avoid ties.

Naïve Bayes (NaiveBayes)

The Naïve Bayes classifier is based on Bayes Theorem which assumes class conditional

independence. In this regard, we refer to the idea that this classifier assumes total independence

from its attributes and uses a probabilistic learning to classify instances. It calculates the

probability that an instance is in a certain class given a certain feature, and does this for all features.

Support Vector Machines (SVM)

Support Vector Machines, or SVM for short is a classifier based on regulating the data

following decision planes designed by decision boundaries. The Support Vector Machine classifier

attempts to minimize the expected empirical loss on the training data, it works under the

probabilistic assumption taken from previous examples. Essentially, SVMs create a maximum

margin separator in the attempt of creating a decision boundary with the largest possible distance

to the previous examples in order to generalize the next state.

**Comparison of Results**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Parameters (Yes/No)** | | | |
| **Test #** | **Tokenization** | **Stop Word Removal** | **Stemming** | **Attribute Selection** |
| 1 | No | No | No | No |
| 2 | Yes | No | No | No |
| 3 | Yes | Yes | No | No |
| 4 | Yes | Yes | Yes | No |
| 5 | Yes | Yes | Yes | Yes |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Accuracy of Tests (%)** | | | |
| **Test #** | **Naïve Bayes** | **SVM/SMO** | **Decision Trees/ J48** | **KNN** |
| 1 | 48.4371 | 86.3071 | 79.7234 | 37.4136 |
| 2 |  |  |  | 40.1522 |
| 3 |  |  |  | 37.4136 |
| 4 |  |  |  | 37.4136 |
| 5 |  |  |  | 44.2462 |

Test 1 is computed on the provided semeval\_twitter\_data.arff file, and uses the StringToWord attribute filter in Weka to extract words. Complete results can be found in the following files:

* test1\_naivebayes.txt
* test1\_svm.txt
* test1\_decisiontrees.txt

Test 5 kept only the attributes of:

Category, Can, Day, Friday, Good, I, Saturday

Rough draft above