Group Assignment

INFO284 / Machine Learning

Assignment 1

Candidate numbers:

90 | 36 | 18 | 107 | 64

DEPARTMENT OF INFORMATION SCIENCE AND MEDIA STUDIES

UNIVERSITY OF BERGEN

Spring 2018

The performance of the program is satisfying. To run the entire set through our program took between 36-39 seconds. When we ran our naive-bayes algorithm, it predicted 19280 of 25000 files correctly, this is a prediction rate of approximately 78%. We tested this on a Macbook Pro, with SSD and fast reading/writing speed, running in the recent version of PyCharm. Speed may vary when loading files using another computer, but the speed is satisfying enough when it comes to run on slower machines, so the difference between a fast and a slow machine should be miniscule. Since the point of this assignment is to have a program that manages the highest point of prediction rate, within a reasonable amount of time. Having a success rating laying in the high 70 percents, with less than a minute of running time. We feel that we've managed to build a good machine learning program, that performs excellent when it comes to performing the task given.

We ran into some issues when it came to an earlier iteration of the program, which made a higher failure rate the more of the training set you put into it. But after sorting out the kinks and issues the program finally worked to our expectations. After getting the review of the entire program, both during labs and during the first hand inn. We managed to rewrite which made the program better.

When it comes to this type of machine learning, it's important to build up a vocabulary of words for the program to use. Figuring out which words are positive and negative. The difficulty of this is to have a vocabulary big enough to fit enough words, but also slim enough so the program runs as fast as we'd like it to.

The smoothing technique could be implemented to make the program work better with the training set, which would make the success rate higher than it is today. When it comes to filter out unknown words that might appear in the test set that the classifier might not known the probability of. A filter could also be implemented, going through neutral words and excluding them from being used during the test phase. It's also difficult to figure out words that are used to empathise other words, such as incredible. Which can be used both in a positive and a negative review. When it comes to the train set not having any neutral reviews, so the program has no idea when it has to guess from the test set. When a review is closer to being neutral, the program has to guess if it's positive or negative neutral often resulting in a wrong guess.

Porter stemming algorithm is something that also can be implemented to reduce time for the program. It might not improve the overall success rate, but would decrease the overall time of the program. Porter stemming works that it takes regular english words and removes the endings of some while words such as possible and possibly gets worked into one. Reducing the overall word count for the vocabulary while maintaining important words.

N-grams are another possible addition to a machine learning program using text. This is another way of parsing words together, having the possible outcome to guess which word might follow another word. The difficulty of this is adding such a complex algorithm, which in most cases adds to the overall time the program takes to run. But this is another example on how to improve the machine learning program.

Yet another addition to the machine learning program is the use of adding Word Frequency, this is another addition to the BagOfWords algorithm. But the issue here is to filter out the highest frequency of words, such as “the”, “a”, “and”. Which has no use to learn about something positive or negative. Using Words Frequency each words get a value between 0 and 1, depending on how much they appear inside the given training set.