DS-222 Assignment 1

Aakash Khochare(14050) [*1]

Abstract

As a part of this assignment, the training and testing of a Naive Bayes Classifier for classification of web documents was implemented in Java and also the Hadoop Map Reduce framework, and analysed for it's scalability characteristics.

1. Introduction

The Naive Bayes Classifier uses the conditional probability of the occurance of a term given a class and the probability of the class occuring, to determine the probability of the class given the term and also assumes conditional independence between terms.

Training this classifier essentially becomes a problem of counting the co-occurences of terms with classes. A single threaded Java program was implemented as the local version for determining these counts. The output of the Java program was just the required counts. The algorithm was exactly the one as mentioned in the slides. In the testing phase, a slightly modified formula was used.

In Map Reduce implementation, the training task was done in one Map-Reduce task, while the testing requires two Map-Reduce cycles. The output of the Map-Reduce tasks are similar in format to the output from the local implementations except that they are stored in the HDFS.

2. Analysis

Since the two implementations in their training phase only count up values, they both have the same testing accuracy of 62.57%. The number of parameters in a Naive Bayes model is given by $vk + k - 1$ where v is the size of the vocabulary and k is the number of classes. In the training dataset, $v = 512278$ and $k = 50$, so the number of parameters are $2,56,13,949$.

The results for weak scaling of Training by varying the number of Reducers is shown in Figure 1. We can see that the time does reduce almost linearly as we increase the number of reducers to 8. However, at 10 reducers the time increases, which can be attributed to the increase in the message passing in comparison to the actual compute.

The results of weak scaling for Testing by varying the number of Reducers is shown in Figure 2. This seems to follow a wierd saw tooth like behaviour. However, testing on a larger dataset could possibly help in understanding the behaviour better.

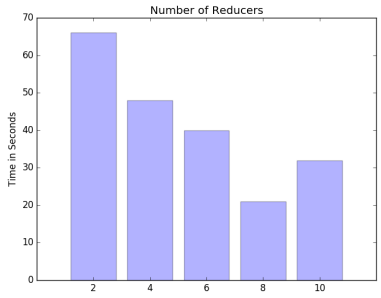

Figure 1. Weak Scaling for Training

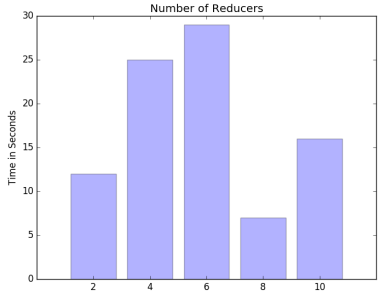

Figure 2. Weak Scaling for Testing

[1]Indian Institute of Science, Bangalore. Correspondence to: Aakash Khochare <aakashkhochare@gmail.com>.