DS 222: ML with Large Datasets (2017)
Assignment 1

Tony Gracious [*] [1]

Abstract

This is the report experiments on Naive Bayes implementation in local and map reduce mode

1. MapReduce Model Description

1.1. Training Stage

- Stage 1: The first map function will preprocess the text by removing punctuations, html tags and normalizing text to lower case. Then it will print out parameters as key and 1 as value as said in the slides in the class. The reduce stage will find the frequency of these parameters. Each line of the output of this stage is like any of the following $Y = ANY\ count1$, $Y = Class\ count2$, $Y = Class \wedge X = word\ count3\ andY = Class \wedge X = ANY\ count3$

- Stage 2: This map stage will use the previous output of the reducer and filter output only parameters with word of text is parameters and the reduce stage will combine the different parameters for same word. The each line of the output of this stage is like this $word\ class1\ count1\ class2\ count2, class3\ count3$

- Stage 3: This stage will filter the out parameters that will common to all text samples like from first map output $Y = ANY\ count1$, $Y = Class\ count2$, $Y = Class \wedge X = ANY\ count3$. As this is small number of parameters, this will be downloaded and stored in local node and distributed among the worker node while testing. This is a map only stage

- Stage 4: This stage will find all unique words using the output from stage2. This will use a dummy key to all words from stage 2 and use a single reducer to output the sum.

- A local python script will read output from stage 4 and file downloaded from Stage 3 and add no of unique words and no of classes to it. This file will distributed among the worker nodes while testing. This will have a total of 103 lines of text.

1.2. Testing Stage

- Stage 1: This stage give each sample in the test file an positive integer ids. The map is identity mapper and as we have to add numbers incrementally single reducer must be used.

- Stage 2: The output from stage 1 is used as input. This stage output the words with corresponding sample id with tilde add to each ids

- Stage 3: This stage will take output of Stage 2 of training and Stage 2 of testing as input and output this stage will word, id and corresponding frequency among the classes

- Stage 4: This combine output from stage 3 based on id as key. So the output of this stage will be sample id along with class distribution of the words in the id.

- Stage 5: This stage will do the prediction. This will use the output from Stage 4 and Stage 3 which now locally available will distributed among the worker nodes. The is map only stage which will calculate logliklihood for each samples in the test and output prediction class

- Stage 6: This stage will use Stage 1 output and extract the true class.

- Stage 7: Uses the stage 5 and stage 6 as input and will output prediction as true or false for each samples

- Stage 8: This uses stage 7 and calculate the accuracy

Total are 1089688 parameters and of which 103 of them are cached distributively.

2. Local Naive Bayes

Implemented as discussed in the class. Total number of parameters is 1089688

3. Experimental Results

The table 1 has been created by measuring the training time (using trianing dataset)and testing (testing dataset) us-

Table 1. Timing

Method	Training time	Testing time
MapReduce (R = 2)	4m25.535s	6m15.107s
MapReduce (R = 4)	5m14.485s	7m26.854s
MapReduce (R = 6)	3m55.325s	6m12.573s
MapReduce (R = 8)	3m33.234s	6m12.165s
MapReduce (R = 10)	3m24.248s	6m28.224s
Local	0m41.99s	3m31.47s

Table 2. Classification accuracy

Method	Training	Testing	Development
Local	95.32	94.53	94.14
MapReduce	95.3	81.03	76.32

ing the `time` command in the linux terminal.We can see as the reducer count is reduced considerably training time and testing time increased due as reducers have go through more data in streaming manner and this take time. We can also observe that If the increase in number of reducers is small the gained computation cost may not over come the communication cost of sending reducers to different nodes. Another observation is that the implementation on local machine is faster that distributed as all computations are done in memory with very less disk reading operations

The testing accuracy in 2 is calculated by combining training and development data together for training. You can observe there is discrepancy in local and mapreduce so there are some more bugs in the mapreduce implementation which needs more testing to figure it out.

The reducer timing 3 in the table is measure for stage 1 of the training by measuring the time between mapper 100% reducer 0% to mapper 100% reducer 100% on training dataset. This measurements been used for plotting the figure 1. There is a linear decrease in time taken for reduce operation as number of reducers increases as the data handled by each reducer decreases as the number of reducers increases.

Acknowledgements

I would like acknowledge the help of Bhutesh for helping me to find the path of hadoop streaming api and also in the debugging the error on distributed caching of common paramters file obtained in Stage 3 of training stage. I also like to acknowlege the help of this `http://www.michael-noll.com/tutorials/writing-an-hadoop-mapreduce-program-in-python/`. The word counting code from this website has been used and modified for creating different mapper and reducer for the naive bayes map reduce implementation. Another website of help was `http://mlwhiz.com/blog/2015/05/09/Hadoop_Mapreduce_Streaming_Tricks_and_Techniques/` this helped me in showing how to distributed caching of files.

Table 3. Reducer Time

Count	time
2	79sec
4	52sec
6	58sec
8	32sec
10	22sec

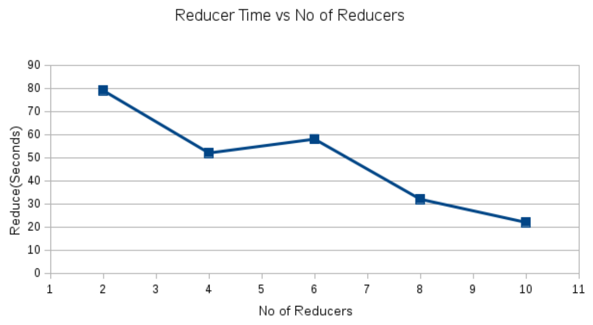

Figure 1. Reducer timing vs No of Reducers

References

Langley, P. Crafting papers on machine learning. In Langley, Pat (ed.), *Proceedings of the 17th International Conference on Machine Learning (ICML 2000)*, pp. 1207–1216, Stanford, CA, 2000. Morgan Kaufmann.