Project
Data Mining: Learning from Large Data Sets
FS 2015

LAS Group, Machine Learning Institute
Dept. of Computer Science, ETH Zürich
Prof. Dr. Andreas Krause
Web: http://las.ethz.ch/courses/datamining-f15/
Email questions to:
dm2015@lists.inf.ethz.ch

Project 1, Sep 28, 2015
Locality Sensitive Hashing

You should not use any other data other than those that we provide you. You are also not allowed to hand-label the given data.

1 Introduction

In this project we are going to consider an application of Locality Sensitive Hashing - detecting duplicate videos. Suppose that there exists a company which offers a web service that allows users to upload various (cat) videos. When an user uploads a copyrighted video, the company is sooner or later sued by the copyright owner. In order to solve this expensive problem, the company employed you. Your task is to develop an efficient method to to detect whether a video is a near-duplicate of a copyrighted video, in which case you don't want to allow the user to publish it.

Your task is to develop an efficient method to to detect whether a video is similar to a copyrighted video, in which case you don't want to allow the user to publish it.

2 Input and Output Specification

You are given two text files: "train.txt" and "duplicates.txt". Each line of the training dataset contains the features for one video file and is formatted as follows: string "VIDEO XXXXXXXXX" followed by a list of space delimited integers in range $[0, 10000]$. You can consider them equivalent to shingles in the context of near-duplicate document retrieval. Furthermore, we will consider the Jaccard similarity based the video features. Duplicates file contains the true near duplicates (documents that are at least 85% similar). Each line is a tab-delimited pair of integers where the first integer is always smaller. Note that similarity is a symmetric function.

You are asked to provide a Map function and a Reduce function written in Python. The Map function receives files (formatted as "train.txt") as the standard input. The output of the Reduce function should be tab separated list of integers representing the duplicated documents your LSH based algorithm found. The number of hash functions used has to be less than or equal to 256. We have provided some template code for both the Map and Reduce function as well as a function to check the F_1 score of your submission on the local training dataset. Each submission on the project website will be run against a bigger dataset that contains many more instances than the training dataset but the sample given to you is representative, in a sense that you can tune the parameters of your algorithm locally (up to a certain degree). The number of submissions per team is limited to 50. The submission with the highest F_1 score will be used for grading.

3 Evaluation and Grading

The evaluation metric that we are going to use is F_1 score. For each line of the output of your MapReduce program we will check whether two documents reported on that line are in fact at least 85% similar (we have the ground truth for the larger dataset). If they are this row will count as one true positive. If they are not it will

count as one false positive. In addition, each document pair that is similar but was not reported by your algorithm will count as one false negative. Given the number of true positives, false positives and false negatives, TP, FP and FN respectively, we can calculate:

1. precision, also referred to as Positive predictive value (PPV), as $P = \frac{TP}{TP+FP}$
2. recall, also referred to as the True Positive Rate or Sensitivity, as $R = \frac{TP}{TP+FN}$.

Given precision and recall we will calculate the F_1 score defined as $F_1 = 2PR$. Your task is to maximize this We will compare the F_1 score of your submission to two baseline solutions: a weak one (called "baseline easy") and a strong one (called "baseline hard"). These will have the F_1 of FBE and FBH respectively, calculated as described above. Both baselines will appear in the rankings together with the F_1 score of your solutions. The grade is computed as the *maximum* of the following two percentages.

Performing better than the weak baseline will give you 50% of the grade, and matching or exceeding the strong baseline on the test set will give you 100% of the grade. This allows you to check if you are getting at least 50% of the grade by looking at the ranking. If your F_1 score is in between the baselines, the grade is computed as:

$$grade = (1 - \frac{FBH - F_1}{FBH - FBE} \cdot 50\% + 50\%.$$

3.1 Report

You are requested to upload a ZIP archive containing the team report *and* the code. We included a template for LaTeX in the file `report.tex`. Please keep the reports brief (under 2 pages). If you do not want to use LaTeX, please use the same sections as shown in `report.pdf`. The reports are uploaded on the same page as the test set submissions.

3.2 Deadline

The submission system will be open from **Monday, 28.09.2015, 17:30** until **Sunday, 11.10.2015, 23:59:59**.