Data Mining: Learning From Large Data Sets

Fall Semester 2015

{usamuel, adavidso, kurmannn}@student.ethz.ch

November 29, 2015

Extracting Representative Elements

Problem formulation. Given a set of features of images that belong to different (unknown) clusters, the goal was to find representative elements for each cluster. The quality of our selection is quantified by the sum of quadratic distances that each representative has to the elements of its cluster. This problem statement is equivalent to the solution of k-means, but we were able to use a map-reduce framework to speed up this process.

Approach and Results. To solve this problem, we used SciKit Learn's integrated implementation of kmeans, which uses kmeans++ to determine starting centers and local minima.

In our initial approach, we ran kmeans on the mappers to compute k=200 centers each, which were then all passed to the reducer, who in turn ran kmeans to reduce the number of centers to 100, which we then proceeded to output as our result. Using the right number of centers for the mappers and the reducer gave us the surprisingly high score of 9.17742.

This was without taking into the consideration the weights of the clusters/coresets passed by the mappers. We tried to override SciPy's kmeans update centers method, but we failed because it is defined in the binary file `_k_means.x86_64-linux-gnu.so`.

In our second approach, we kept the mappers mostly the same, but we tried to take into account the weights of the centers outputted by the mappers. For this, we stopped performing kmeans in the reduce step, but instead opted for merging the points manually. For this, the reducer selected a random center and merged it with the closest other center by taking a weighted average. This approach netted us with a score of 9.02585.

Workload distribution.