Mining Massive Data
Programming Assignment 1

Lukas Hinterleitner, Lukas Till Schawerda, Kilian Wohlleben

universität wien

Implementation

The aim of this assignment was to implement a model for predicting the music genre of a song based on locality sensitive hashing. The implementation utilizes python *class* objects to perform the required tasks in an easily accessible way. First, a class *HashTable* is created which puts into practice locality sensitive hashing based on random projections as described in the assignment. Then, a class *LSH* is used to compute locality sensitive hashing for the specified number of hash tables. Finally, the class *ApproximateNearestNeighbour* takes care of both the training of the model and the prediction of new data points. Potential hyperparameters were preselected in an effort to achieve good performance. This was consequently verified by training the model with these parameters using the training data and then considering the prediction accuracy on the validation data. The specific hyperparameters we tested are listed in the *Results* section of this report.

If there are less than k nearest neighbors in the same bucket as a song query, the prediction is made based on the nearest neighbors that are still available. This choice was made to ensure a reasonable prediction in as many cases as possible.

Results

We tested these hyperparameters during the model training process:

Parameter	Description	Tested values
l	Hash length	50, 100, 150
n	Number of hash tables	10, 15, 20
k	Number of nearest neighbors	10, 15, 20
m	Distance measure	euclidean, cosine

The results they achieved can be found by running the code, as they are conveniently printed into the console while running the training process.

l	n	k	m	Validation accuracy
50	10	10	euclidean	0.3215
50	10	10	cosine	0.3091
100	15	15	euclidean	0.3157
100	15	15	cosine	0.3132
150	20	20	euclidean	0.3333
150	20	20	cosine	0.3190

In the end, we settled on the parameters l=150, n=20, k=20, and m=Euclidean. Selecting these parameters leads to a classification accuracy of 0.3212 on the test set.

Comments

As described in the assignment, the random matrix **R** is generated according to a procedure described by Achlioptas (2003), where each element is drawn as

$$r_{i,j} = \sqrt{3} \cdot \begin{cases} +1 & \text{with probability } \frac{1}{6} \\ 0 & \text{with probability } \frac{2}{3} \\ -1 & \text{with probability } \frac{1}{6}. \end{cases}$$

This is beneficial compared to drawing $r_{i,j}$ from a Gaussian distribution because it leads to a sparse matrix where many entries are equal to zero. This in turn makes matrix multiplication computationally easier. Bingham and Mannila (2001) show that this approach displays similar behavior to Gaussian distributed entries.

The runtime of our algorithm will likely be longer than the runtime of exact nearest neighbor search. This can be explained by the fact that we additionally need to compute hashing etc., which would not be necessary for an exact algorithm. However, for very large data sets, exact nearest neighbor search is no longer feasible because of the massive amount of comparisons that need to be computed, as for each instance we will have to compute no_train_samples² distances, while for LSH we only need no_train_samples * n computations, so it scales linearly. In this setting, our algorithm and locality sensitive hashing in general will likely display a better runtime.

We spent about 8 hours per person working on this assignment. Lukas wrote most of the code, Till did research and worked on the report, Kilian evaluated the results and worked on the report.

Sources

- Bingham, E., Mannila, H. (2001, August). **Random projection in dimensionality reduction: applications to image and text data.** In Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining (pp. 245-250).
- Achlioptas, D. (2003). **Database-friendly random projections: Johnson-Lindenstrauss with binary coins.** Journal of computer and System Sciences, 66(4), 671-687.
- Defferrard, M., Benzi, K., Vandergheynst, P., Bresson, X. (2016). **FMA: A dataset for music analysis.** arXiv preprint arXiv:1612.01840.