EE 381V: Large Scale Learning **Spring 2013**

Lecture 1 — January 15

Lecturer: Caramanis & Sanghavi *Scribe: Brian Bates*

1.1 Topics Covered

- What is Big Data
- Nearest Neighbors
- Hashing and Locally Sensitive Hashing
- Amplification

1.2 What is Big Data

There are two main issues in modern data: size and dimensionality. Large data sizes can cause access and storage problems. With high-dimensional data, the big issue is statistical. Namely, empirical distributions do not provide good approximations. The main cure for size issues is parallelization (divide and conquer). With high dimensional data, the solution is either to reduce the dimensionality of the problem or to make structural assumptions about the data.

1.3 Nearest Neighbors

Nearest neighbors is the problem of finding the nearest point in a set to a query point. One variation is k-nearest neighbors. That is, given a set of points, what are the k nearest points to a query point p? Figure 1.1 illustrates the nearest neighbors problem.

Nearest neighbors problems have several applications.

- **Clustering/Classification:** Suppose you are given labels for each point in a set. What is the label of a query point?
- **Recommendation Systems:** Given a user's data, recommend items they might want. This can be done by finding users that are nearby and choosing items that are popular among those users.
- **Document Analysis:** Suppose you are given many document files. How can we group them? If we are additionally given a query document, how can we find a similar

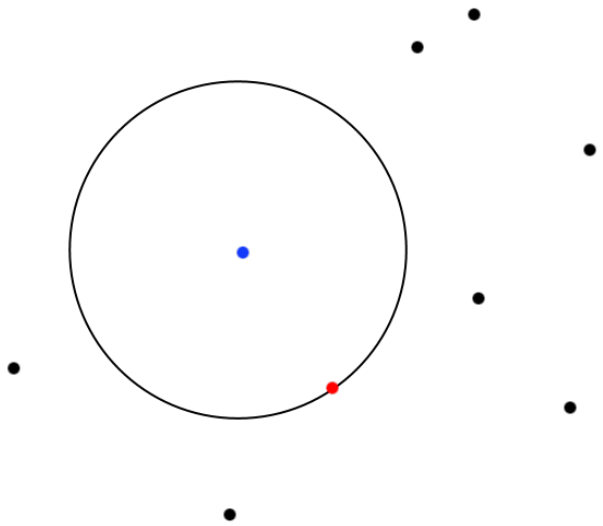

Figure 1.1. Nearest Neighbor. Blue's nearest neighbor is highlighted in red.

document? We can use techniques like Bag of Words to represent the document as a vector, then apply nearest neighbors on the vectors.

- **Vision Applications:** Suppose you have a large set of images of a particular scene. An object is then placed in the scene. How can we remove the object from new images of the scene? For example, authors wanted to remove a house from a scene. The authors used nearest neighbors to find the 1000 closest images and interpolated to generate an image without the house. This is called In-Painting.

Today, we want to look at how to perform nearest neighbors when there are many high-dimensional points (n-points and d-dimensions). The brute force solution is linear time $O(nd)$. There are several other algorithms that are efficient in n, but exponential in d. Therefore, high-dimensionality can be an issue. Shopping in a supermarket provides a good example of a high-dimensional scenario. There are thousands of items in the supermarket. If we were to represent each item as a binary variable (1 meaning purchased), we can represent each shopper by a binary vector of all items in the supermarket. We can see that while each shopper only buys relatively few items, the dimensionality of the problem is still very large.

1.4 Hashing and Locally Sensitive Hashing

Hashing is a method for reducing the dimensionality of data. The process can be seen in figure 1.2. The idea behind a hashing is to find a function h, called a hash function, that maps high-dimensional data into a (relatively) low-dimensional range.

The choice of hash function is important. In order for hashing to be useful, we want to ensure that elements that are close in the high-dimensional domain are still close in the

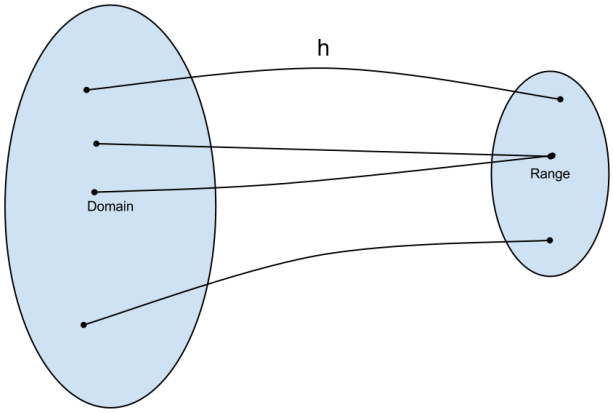

Figure 1.2. Hashing.

lower-dimensional range. This motivates the idea of locally sensitive hashing. In locally sensitive hashing (with a finite range), points close by are more likely to be mapped to the same bucket when compared with points that are far away. After mapping, the distance between buckets is not considered; Points in the same bucket are considered the nearest.

Definition 1. *A family H of hash functions is (r_1, r_2, P_1, P_2)-locally sensitive if for all points p and q, given that $r_1 < r_2$ and $P_1 > P_2$ the following hold:*

$$\|p - q\| \leq r_1 \quad \Rightarrow \quad P_H\left(h(p) = h(q)\right) \geq P_1 \tag{1.1}$$

$$\|p - q\| \geq r_2 \quad \Rightarrow \quad P_H\left(h(p) = h(q)\right) \leq P_2 \tag{1.2}$$

For example, consider a dataset consisting of d-length binary vectors under the Hamming distance. We want to find a function such that $P\left(h(i) = h(j)\right)$ (the probability of points i and j landing in the same bucket) decreases as $d(i, j)$ (the Hamming distance) decreases. The optimal hash function f in this case is obtained by randomly selecting an index k. For every point p in the domain, if $p_k = 1$, then $f(p) = 0$, otherwise $f(p) = 0$.

1.5 Amplification

Suppose H is (r_1, r_2, P_1, P_2)-locally sensitive. Let us consider the following operations.

- **AND:** Pick r functions $h_1, ..., h_r$ iid uniformly at random from H. Define $\tilde{h}$ such that $\tilde{h}(p) = \tilde{h}(q) \Leftrightarrow h_i(p) = h_i(q) \forall i \in 1, ..., r$.
- **OR:** Pick b functions $h_1, ..., h_b$ iid uniformly at random from H. Define $\hat{h}$ such that $\hat{h}(p) = \hat{h}(q) \Leftrightarrow h_i(p) = h_i(q)$ for at least one $i \in 1, ..., b$.

1.6 Next Class

In the next class we will further discuss amplification and its applications. We will also investigate a variation of the nearest neighbors problem called randomized R-nearest neighbors.