THE UNIVERSITY OF TEXAS AT AUSTIN
Department of Statistics and Data Sciences
College of Natural Sciences

SDS 385: Stat Models for Big Data

Lecture 7: Nearest neighbor methods

Purnamrita Sarkar
Department of Statistics and Data Science
The University of Texas at Austin

https://psarkar.github.io/teaching

Nearest neighbor queries

- Many applications need efficient nearest neighbor search
- It can be kernel regression
- Matching and retrieval
- Kernel density estimation

A concrete example: Min hash

- Lets start with a simple setting.
- You have documents which can be represented by sets of words, or shingles, which are none other than moving window of words.
- If a document is ‘This is Stat models for Big data’, then 2-singles are $\{$‘This is’, ‘is Stat’, ‘Stat models’$\}$ etc.
- The goal is to remove duplicate documents.
- For $1M$ documents, doing all pairs of similarity would take about 5 days.

Jaccard similarity

- Consider two sets S_1, S_2
- A common similarity measure is the Jaccard index:

$$J(S_1, S_2) = \frac{|S_1 \cap S_2|}{|S_1 \cup S_2|}$$

- Consider the binary representation of two sets $S_1 = 10111$ and $S_2 = 10011$
    - $|S_1 \cap S_2| = 3$
    - $|S_1 \cup S_2| = 4$
    - Jaccard score $1/4$

Hashing: main idea

- Goal: find a hash function $h(.)$ such that
  - If $sim(C_1, C_2)$ is high, then w.h.p $h(C_1) = h(C_2)$
  - If $sim(C_1, C_2)$ is low, then w.h.p $h(C_1) \neq h(C_2)$
- Not all similarity functions allow such a hash function
- For the Jaccard score however, such a function does exist.

Min Hashing

- Write the document dataset as a binary matrix of shingles by documents
- Consider a permutation π of the elements, or the words, or the shingles or the rows
- $h_\pi(C)$ is the index of the first (in the permuted order π) row in which column C has value 1.
- In other words:

$$h_\pi(C) = \min(\pi(C))$$

- Use many hash functions (i.e. via random permutations) to create a signature of the columns

Example

Permutation π

2	4	3
3	2	4
7	1	7
6	3	2
1	6	6
5	7	1
4	5	5

Input matrix (Shingles x Documents)

1	0	1	0
1	0	0	1
0	1	0	1
0	1	0	1
0	1	0	1
1	0	1	0
1	0	1	0

Signature matrix *M*

2	1	2	1
2	1	4	1
1	2	1	2

Similarities:

	1-3	2-4	1-2	3-4
Col/Col	0.75	0.75	0	0
Sig/Sig	0.67	1.00	0	0

Key claim

- $P(h_\pi(C_1) = h_\pi(C_2)) = J(C_1, C_2)$
- Consider a document X and let $y \in X$ be an element of it.

$$P(\pi(y) = h_\pi(X)) = 1/|X|$$

  - Since it is equally likely for any element to become the smallest element under a random permutation
  - For C_1, C_2 the probability that some element $y \in C_1 \cup C_2$ is the min-hash is $1/|C_1 \cup C_2|$
  - The probability that the two min-hashes are the same is the same as the probability that one of the elements in the intersection is the min-hash, i.e. the probability becomes $|C_1 \cap C_2|/|C_1 \cup C_2|$

Key claim

- The hash function only returns 1 or 0 not a number in $[0, 1]$
- Thats why you need multiple hash functions and take the average
- For 100 random permutations, each document is now represented as a vector in 100 dimensions, so we have compressed the original long vectors intro short signatures while not losing the signal, which is the similarity between documents in this case

Min hashing

- Permuting rows is prohibitive.
- You can use approximate linear permutation hashing.
- $h(x; a, b) = ((ax + b) \mod p) \mod n$ where a, b are random integers and p is some prime number larger than n.

Acknowledgment

Ullman's lecture notes from "Mining of Massive Datasets"