Module 6: Dimensionality Reduction

Lecture 32: Embedding

Prev topic Next topic Prev page
Next page

The Lecture Contains:

Module 6: Dimensionality Reduction

Lecture 32: Embedding

Prev topic Next topic Prev page
Next page

Embedding

- Transformation from one space and distance function to another space (generally vector) and distance function (generally Euclidean)

- Both nature of space and distance function are important

Embedding

- Transformation from one space and distance function to another space (generally vector) and distance function (generally Euclidean)
- Both nature of space and distance function are important
- Example: Given a metric space with distance function d with N objects, embed into N-dimensional vector space with distance function L_∞

Embedding

- Transformation from one space and distance function to another space (generally vector) and distance function (generally Euclidean)

- Both nature of space and distance function are important

- Example: Given a metric space with distance function d with N objects, embed into N-dimensional vector space with distance function L_∞

  - $f(O_i) = \{d(O_i, O_1), \ldots, d(O_i, O_N)\}$

  - Dimension i of a point is dependent on O_i

  - Distance-preserving

  - In general, not possible for other L_p norms

  - Example

    - Original: $O_1 = (2,3), O_2 = (1,5), O_3 = (6,4), d = L_1$

    - Embeddings: $O'_1 = (0,3,5), O'_2 = (3,0,6), O'_3 = (5,6,0), d' = L_\infty$

    - Distance-preserving:

$$d(O_1, O_2) = 3 = d'(O'_1, O'_2)$$

$$d(O_2, O_3) = 6 = d'(O'_2, O'_3)$$

$$d(O_3, O_1) = 5 = d'(O'_3, O'_1)$$

Prev topic Next topic Prev page
Next page

Lipschitz embeddings

- Embedding of database D with metric distance d onto k-dimensional vector space
- Choose k subsets of D
- Each subset A_i is called a reference set
- Define distance of object o to set A as $d(o, A) = \min_{a \in A} d(o, a)$
- Feature vector $f(o) = \{d(o, A_1), \ldots, d(o, A_k)\}$
- In earlier example
    - $A_i = \{o_i\}$ and L_∞
- Contractive under L_1 (after scaling)
- Says nothing about distortion

Proof of contractive embedding property

- Consider any subset A_i
- $\exists a_1 \in A_i,\ d(o_1, A_i) = d(o_1, a_1)$ and $\exists a_2 \in A_i,\ d(o_2, A_i) = d(o_2, a_2)$
- $\therefore\ d(o_1, a_1) \le d(o_1, a_2)$ and $d(o_2, a_2) \le d(o_2, a_1)$

$$|d(o_1, A_i) - d(o_2, A_i)| = |d(o_1, a_1) - d(o_2, a_2)|$$

$$= \max \begin{cases} d(o_1, a_1) - d(o_2, a_2) \\ d(o_2, a_2) - d(o_1, a_1) \end{cases}$$

$$\le \max \begin{cases} d(o_1, a_1) - d(o_2, a_1) \\ d(o_2, a_2) - d(o_1, a_2) \end{cases}$$

$$\le d(o_1, o_2)$$

- Scale $d'(f(o_1), f(o_2)) = \sum_{i=1}^{k} |d(o_1, A_i) - d(o_2, A_i)|$ by $1/k$ to get contractive embedding

Prev topic Next topic Prev page
Next page

LLR embedding

- Linial, London, Rabinovich
- A variant of Lipschitz embedding
- Guarantees $O(\lg N)$ distortion
- $k = O(\lg^2 N)$ subsets
- Each group of $O(\lg N)$ subsets is of size 2^i where $i = 1, \ldots, O(\lg N)$
- Size of subset $A_i, j = i, \ldots, \lceil \lg^2 N \rceil$ is $2^{\lfloor \frac{i-1}{\lg N} + 1 \rfloor}$
- Assume distance function in embedded space is $d' = L_p$
- Embedding is $f(o) = k^{-1/p}\{d(o, A_1), \ldots, d(o, A_k)\}$
- With high probability

$$\frac{c}{\lceil \lg N \rceil} d(o_1, o_2) \leq d'(f(o_1), f(o_2)) \leq d(o_1, o_2)$$

- Scaling by $k^{-1/p}$ is needed to make the embedding contractive

Prev topic Next topic Prev page
Next page

SparseMap

- LLR embedding is still impractical
- For large databases, $\lg N$, $\lg^2 N$ are very large
- Dimensionality of embedded space is large
- For large sized A_i , time to determine the corresponding dimension by computing $d(o, A_i)$ is large

Module 6: Dimensionality Reduction

Lecture 32: Embedding

Prev topic Next topic Prev page
Next page

SparseMap

- LLR embedding is still impractical
- For large databases, $\lg N$, $\lg^2 N$ are very large
- Dimensionality of embedded space is large
- For large sized A_i , time to determine the corresponding dimension by computing $d(o, A_i)$ is large
- SparseMap proposes two practical heuristics
- Compute $\hat{d}(o, A_i)$ by considering some (and not all) $x \in A_i$
  - This is a way to avoid computing all $|A_i|$ distances
  - $\hat{d}(o, A_i)$ is an upper bound of $d(o, A_i)$
- Reduce dimensionality by retaining some (and not all) $A_i s$
  - Greedy resampling: Delete A_i 's that contributes largest to stress
  - Sample only some object pairs when computing stress
- Not contractive any more