Learning Collective Behavior using K means Algorithm

S.KRISHNA PRASAD
2013503514
Problem Statement

- Collective behavior means to understand how the individual behaves in a social networking environment.

- k-means variant algorithm is then implemented for clustering which reduces the time required for clustering.

- The edge-centric structure represents that the extracted social dimensions are definitely sparse in nature.

- This scalable approach effectively used over online network collective.
• **Social dimension extraction:**
  - The latent social dimensions are extracted based on network topology to capture the potential affiliations of actors. These extracted social dimensions represent how each actor is involved in diverse affiliations. These social dimensions can be treated as features of actors for subsequent discriminative learning.

• **Learning Collective Behavior:**
  - The discriminative learning procedure will determine which social dimension correlates with the targeted behavior and then assign proper weights. A key observation is that actors of the same affiliation tend to connect with each other.
Classification:
- From the clusters created, we extract features and use these features for understanding the behavior and form the classifier. This is done using Support vector machine. After learning, the SVM algorithm creates a classifier to understand the actor and classify content, showing a recommendation according to the users.

Prediction:
- By taking input as this social dimensions as features to next algorithm, learning and prediction are carried out. This algorithm is based on linear SVM. The discriminative learning procedure will find out related social dimensions by understanding behavior and then gives a proper label.
K Means Variant Algorithm

- **Step 1:** Select randomly \( k \) pivots in the problem state space.

- **Step 2:** Cluster the data in such a way that similar behavioral users belong to a roughly common group.

- **Step 3:** Use the mean of these clusters to find the new center (pivot).

- **Step 4:** Update the centroid after arrival of new connection request.

- **Step 5:** Repeat steps 2 an 3 until centers do not change.
Input: Data instances \( \{x_i \mid 1 \leq i \leq m \} \), Number of clusters \( k \)
Output: Number of clusters \( \{idx_i\} \)
1. Construct a mapping from feature to instances.
2. Initialize the centroid for new cluster \( \{C_j \mid 1 \leq j \leq k\} \)
3. repeat
4. Reset \( \{MaxSim_i\}, \{idx_i\} \)
5. for \( j = 1: k \)
6. identify relevant instances \( S_j \) to centroid \( C_j \)
7. for \( i \) in \( S_j \)
8. Compute \( \text{sim}(i, C_j) \) of instance \( i \) and \( C_j \)
9. if \( \text{sim}(i, C_j) > MaxSim_i \)
10. \( MaxSim_i = \text{sim}(i, C_j) \)
11. \( idx_i = j \)
12. for \( i = 1: m \)
13. update centroid \( C_{idx_i} \)
14. until change of objective value < \( \varepsilon \)
Tool used

- GNU Octave for Windows, Linux etc. binary versions is a high-level interpreted language, primarily intended for numerical computations. It provides capabilities for the numerical solution of linear and nonlinear problems, and for performing other numerical experiments.

- It also provides extensive graphics capabilities for data visualization and manipulation.

- The Octave language is quite similar to Matlab so that most programs are easily portable.

- Octave is written in C++ using the C++ Standard Library.

- Octave runs on various Unices—at least Linux and Solaris, Mac OS X, Windows and anything you can compile it on.
References


