Affinity Propagation

Shenghong Dai
sdai37@wisc.edu

Cheng Li
cli575@wisc.edu

Aditya Barve
asbarve@wisc.edu

December 2019

Abstract

Clustering is a useful technique for data summarization and aggregation which involves dividing the data into groups with some common characteristics. In a previous lecture, we have learned the basic concept of clustering and k-means, a traditional clustering algorithm. While computationally efficient, k-means has its limitations. To address them, we introduce a cluster algorithm called Affinity Propagation (AP), which was proposed in Science 2007 [1]. AP uses iterative message passing between data points to converge to a solution. We explain the intuition behind this process and compare its performance with k-means.

1 Learning Objectives

- Understand the concept of AP clustering
- Be able to decide when to use AP rather than k-means
- Be able to explain the iterative solution for AP
- Be able to evaluate the performance of AP relative to k-means

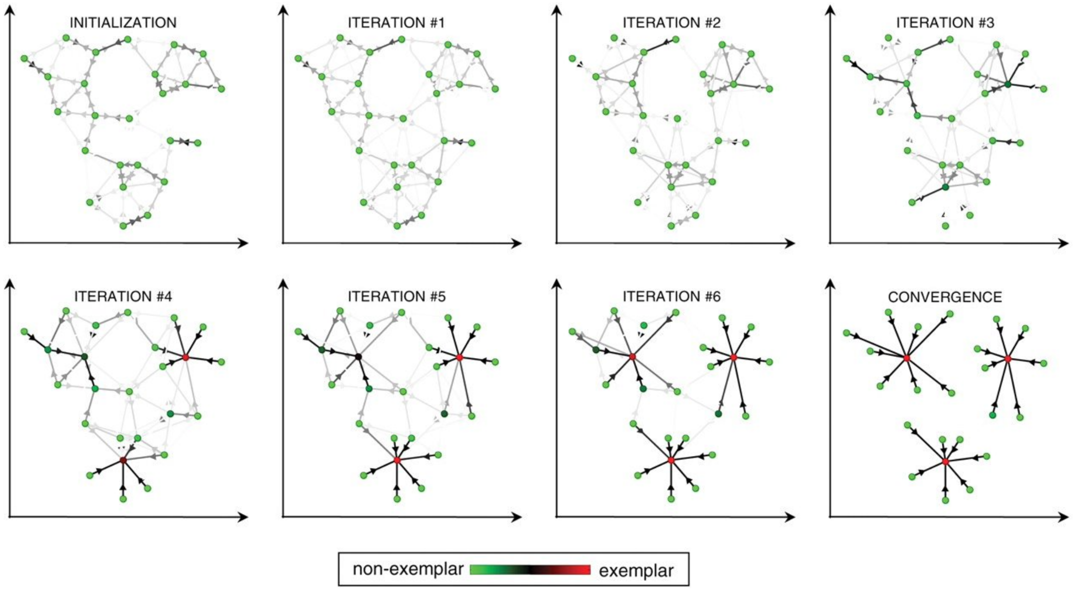


Figure 1: How Affinity Propagation works

2 Background

• Introduction

Clustering data based on a measure of similarity is a critical step in data analysis for many scientific domains and commercial applications. In unit 3, we studied clustering with **k-means**. Each cluster is represented by the "mean" of data points belonging to that cluster. Since these "means" usually do not coincide with known data points, they can be difficult to understand. An alternative strategy is to choose a data point from each cluster which best represents that cluster. When known data points are chosen to represent clusters, these points are called **exemplars**.

We motivate and introduce a different clustering algorithm called Affinity Propagation which uses exemplars rather than "means". AP has been used to partition protein interaction graphs [2], cluster text [3] and categorize unlabeled images [4]. It has been implemented in many popular languages like Python [5], R [6] and Julia [7].

• Limitations of k-means [8]

1. Estimating "k"
   K-means requires the number of clusters "k" as an input parameter. Since clustering is used for unsupervised learning, a good "k" is rarely known beforehand. In practice, the algorithm must be run repeatedly assuming different "k".
2. Sensitive to initial cluster centers
   A bad initial choice could lead to a bad final solution. In practice, this is counteracted by choosing the best solution after repeated runs with different random initialization.

In practice, we run k-means repeatedly, combining different initial cluster centers

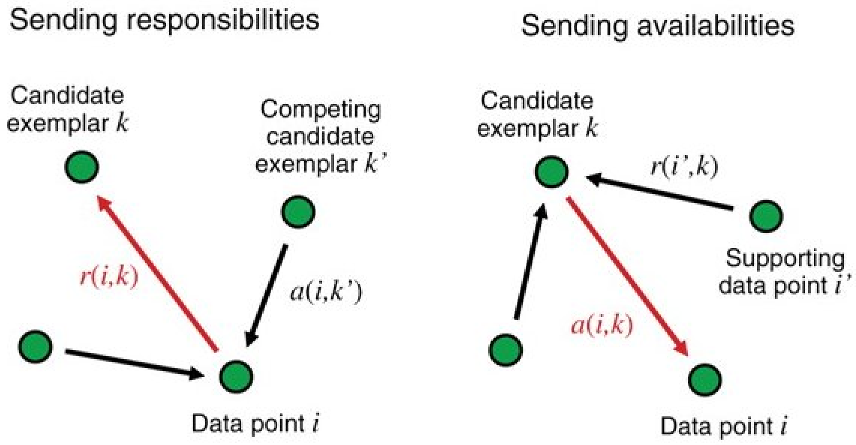


Figure 2: Message passing during Affinity Propagation

with different "k", and choosing the best solution. This quickly becomes expensive as data size increases.

- **Overcoming k-means limitations using AP**

  To overcome k-means' limitations, we introduce a clustering algorithm called Affinity Propagation. Affinity is a measure of similarity between two data points. Propagation means sending messages between data points.

  Initially, k-means chooses "k" random cluster centers. On the other hand, AP assumes all data points are equally likely to be exemplars. Hence **AP does not randomly bias the solution during initialization**.

  To run k-means, we must specify "k". To run AP, we specify a parameter called "preference" which creates more or less clusters depending on the data distribution. Thus **AP provides some flexibility in terms of number of clusters produced**.

- **Further benefits of AP**

  1. AP employs a message passing strategy which is efficient for sparse graphs.
  2. AP can perform asymmetric distance clustering while k-means cannot.
     For example, flying from NYC to London takes longer flying from London to NYC because of "jet stream" winds. Hence the similarity between NYC and London based on flight time is asymmetric. Since the "mean" in k-means is calculated using Euclidean distance which is symmetric, k-means cannot be used to cluster data with an asymmetric similarity matrix.

- **Message passing in AP**

  Recall in the PageRank assignment, we calculated the importance of nodes in a graph by passing real-valued messages between neighboring nodes until their probabilities converged. Similarly, **the key idea of Affinity Propagation is to converge on a set of exemplars by passing "affinity" messages among neighboring nodes**.

AP identifies exemplars through an election process using two types of real-valued messages. It is analogous to electing human leaders among groups of people. First, candidates publicize their willingness to lead (called availability propagation). Next, everyone votes for their favorite candidates (called responsibility propagation).

1. Responsibility propagation

   All nodes send responsibility messages to each of their neighbors. A responsibility $\boldsymbol{R(i,k)}$ is sent from $\boldsymbol{x_i}$ to $\boldsymbol{x_k}$ saying **"I want you to be responsible for leading me"**. A high $\boldsymbol{R(i,k)}$ means $\boldsymbol{x_i}$ prefers $\boldsymbol{x_k}$ to be its leader compared to its other neighbors.

2. Availability propagation

   All nodes send availability messages to each of their neighbors. An availability message $\boldsymbol{A(i,k)}$ is sent from $\boldsymbol{x_k}$ to $\boldsymbol{x_i}$ saying **"I am available and I want to be your leader"**. A high $\boldsymbol{A(i,k)}$ means $\boldsymbol{x_k}$ is confident that it would be a good leader for $\boldsymbol{x_i}$.

If a node has many followers directing high responsibility toward it, then that node will have high availability. However, if a node has few or no followers, then that node's availability will decrease and during the next responsibility propagation round, those followers would direct their responsibility toward a different node with higher availability. After some iterations, some leaders (or exemplars) will emerge. Below are the mathematical formulae for the two types of propagation.

- $\boldsymbol{R(i,k) \leftarrow S(i,k) - \max_{k' \; s.t. \; k' \neq k} \{A(i,k') + S(i,k')\}}$
- $\boldsymbol{A(i,k) \leftarrow \min\{0, R(k,k) + \sum_{i' \; s.t. \; i' \notin \{i,k\}} \max\{0, R(i',k)\}\}}$
- $\boldsymbol{A(k,k) \leftarrow \sum_{i' \; s.t. \; i' \neq k} \max\{0, R(i',k)\}}$

For each data point $\boldsymbol{x_i}$, we find its exemplar data point $\boldsymbol{x_k}$ that maximizes the sum of availability and responsibility $\boldsymbol{\max_k A(i,k) + R(i,k)}$. If i = k, then $\boldsymbol{x_i}$ is its own exemplar.

- **AP Inputs**
  - **Similarity matrix**

    The similarity matrix records pairwise similarity between nodes in the graph. A common similarity metric is negative Euclidean distance.

    $$\boldsymbol{S(i,j) = -||x_i - x_j||_2^2}$$

  - **Preference**

    Preference is a real number corresponding to each node in the graph. $p(i)$ indicates node i's preference toward becoming an exemplar. Often, we set a single preference value for all nodes, meaning they are equally likely to be exemplars at initialization. By varying this common preference value, we can influence the number of clusters produced without specifying the exact number of clusters. If we set the common preference value to an average (like mean or median) of all pairwise similarities, then AP will produce a moderate number of clusters. If we wish to produce fewer clusters, we can choose a preference value lower than the average. Similarly, we can increase the number of clusters produced by choosing a preference value higher than the average.

- **Damping factor**

  Damping factor ($0 < \lambda < 1$) decides the size of the steps we take toward a minima. It plays a role similar to step size(τ) in gradient descent. If τ is too large, we may overshoot the minima and oscillate around it without triggering our termination condition. Similarly, if λ is too small, we may see oscillations around a minima. To avoid oscillations, we dampen or lessen the effect of our propagation updates by increasing λ for smooth convergence to a minima.

Implementation details

We terminate AP when one of the following occurs:

1. After a fixed number of iterations.
2. Exemplars do not change.
3. Performance metric stabilizes.

To evaluate the clustering performance of AP and k-means, we may use ASD (average squared distance) or SSE (sum of squared errors).

Finally, let us compare one iteration of AP and k-means.

AP	K-means
1. Propagate responsibility messages 2. Propagate availability messages 3. Check terminating condition	1. Assign data points to “means” 2. Recompute “means” 3. Check terminating condition

Ethical Concerns

- **Parameter tuning**

  AP allows for setting different preference values for individual data points such that points with a higher preference value are more likely to be chosen as exemplars. Choosing preference values to trade off equity for equality [9] [10] could raise ethical concerns. Even with a common preference value, the damping factor could result in a very different set of exemplars being chosen.

- **Similarity metric**

  Because clustering involves grouping unlabeled data points, choosing a fair similarity metric chosen plays a crucial role in the clusters which emerge. This is especially important when the data points are people. We would need to ensure that the similarity metric captures the relevant social, economic and political factors.

- **Flexible number of clusters**

  Most organizational hierarchies require have a predetermined number of representatives. For example, the United States has 50 governors - no more, no less. In such cases, AP's flexibly choosing the number of exemplars has the potential to disrupt existing systems.

3 Warm-up

(a) Why might we prefer to use AP rather than K-means?

A. AP can handle different types of similarity (including asymmetric similarity).

B. AP does not require you to specify an exact number of clusters.

C. AP is efficient for large, sparse graphs.

D. AP can cure cancer.

(b) Which parameters need to be determined or estimated to run AP? How does each parameter affect the clustering results?

(c) What is the difference between responsibility and availability?

4 Main Activity

Download and run the provided MATLAB scripts and answer the following questions.

*(a) Load the two dimensional dataset using **load('A.mat')**. Without a priori information about the data, how would you decide "k" to run k-means? Try out a few values of k. Now, use AP to cluster the data with damping factor $\boldsymbol{\lambda}$ = 0.5 and each initial preference equal to mean of similarity matrix. Hint: Run the command **ap(A,1,0.5)**. How many clusters does AP produce? Compare AP with k-means and explain AP's advantage in determining the number of clusters.*

*(b) The script **evaluate.m** compares AP with k-means. Run it three times and record both AP and k-means' performance for each run. Does AP's performance fluctuate? Does k-means' performance fluctuate? What difference do you see? Can you explain Why?*

*(c) In ap.m, uncomment line 62 (iteration print line). Run the AP command (**ap(A,1,0.1)**) with a damping factor $\boldsymbol{\lambda = 0.1}$. Does AP converge to a solution? Can you explain why? What would you change so that AP will converge?*

*(d) Use **visualization.m** to help tune the preference. In general, preference is equal to the mean of the similarity matrix. We scale this preference with a coefficient called the preference coefficient. Run **visualization.m** with different values of preference coefficient. How does scaling the preference impact the number of clusters? Hint: Similarity matrix contains negative entries. What happens if you set preference to 0?*

References

[1] Brendan J. Frey and Delbert Dueck. Clustering by passing messages between data points. *Science*, 315(5814):972–976, 2007.

[2] James Vlasblom and Shoshana J. Wodak. Markov clustering versus affinity propagation for the partitioning of protein interaction graphs. *BMC bioinformatics*, 10:99–99, Mar 2009. 19331680[pmid].

[3] R. Guan, X. Shi, M. Marchese, C. Yang, and Y. Liang. Text clustering with seeds affinity propagation. *IEEE Transactions on Knowledge and Data Engineering*, 23(4):627–637, April 2011.

[4] D. Dueck and B. J. Frey. Non-metric affinity propagation for unsupervised image categorization. In *2007 IEEE 11th International Conference on Computer Vision*, pages 1–8, Oct 2007.

[5] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[6] Ulrich Bodenhofer, Andreas Kothmeier, and Sepp Hochreiter. Apcluster: an r package for affinity propagation clustering. *Bioinformatics*, 27:2463–2464, 2011.

[7] Clustering.jl. https://github.com/JuliaStats/Clustering.jl. Accessed: 2019-12-11.

[8] Joaquín Ortega, Ma Rocío, Boone Rojas, and María García. Research issues on k-means algorithm: An experimental trial using matlab. *CEUR Workshop Proceedings*, 534, 01 2009.

[9] Morton Deutsch. Equity, equality, and need: What determines which value will be used as the basis of distributive justice? *Journal of Social Issues*, 31(3):137–149, 1975.

[10] Elizabeth A. Mannix, Margaret A. Neale, and Gregory B. Northcraft. Equity, equality, or need? the effects of organizational culture on the allocation of benefits and burdens. *Organizational Behavior and Human Decision Processes*, 63(3):276 – 286, 1995.

APPENDIX: SOLUTION

Warm-up

(a) ABC

(b) To run AP, we must specify two parameters - preference and damping factor. Preference controls how many exemplars (or prototypes) are chosen. The damping factor reduces the magnitude of responsibility and availability propagation updates. This can help avoid numerical oscillations due to large updates.

(c) A node sends out responsibility messages to tell its neighbors how much it relies on them. On the other hand, a node sends out availability messages to tell its neighbors how much they can rely on it.

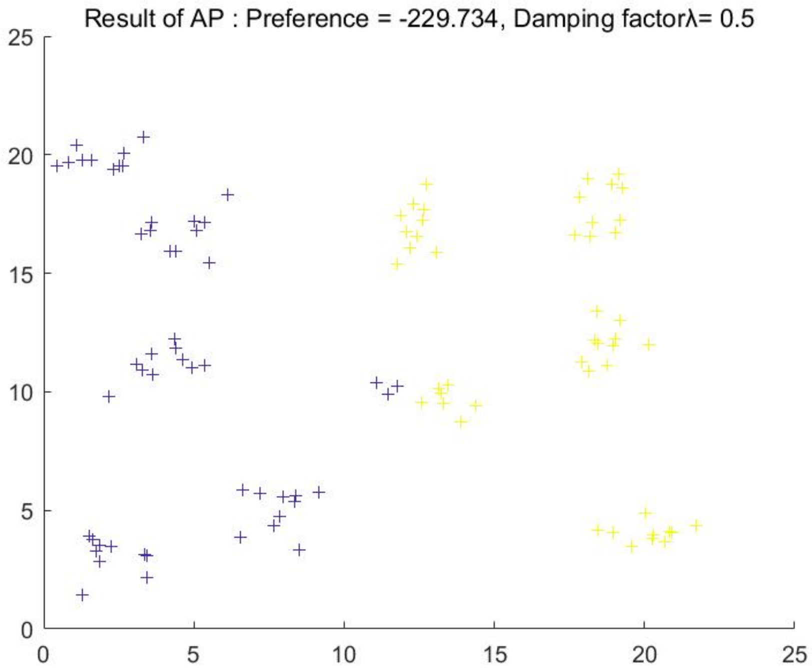
Main Activity

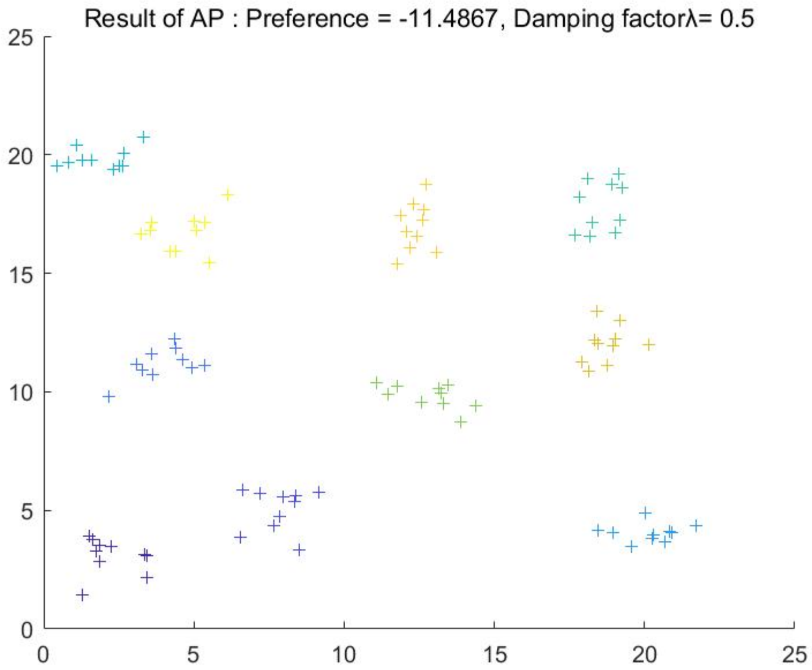
(a) Try multiple values for the number of clusters and then evaluate solutions and pick one that represents the best description of the data.
AP produces 10 clusters.
Unlike k-means clustering algorithm, affinity propagation does not require the user to specify the number of clusters k to be generated.

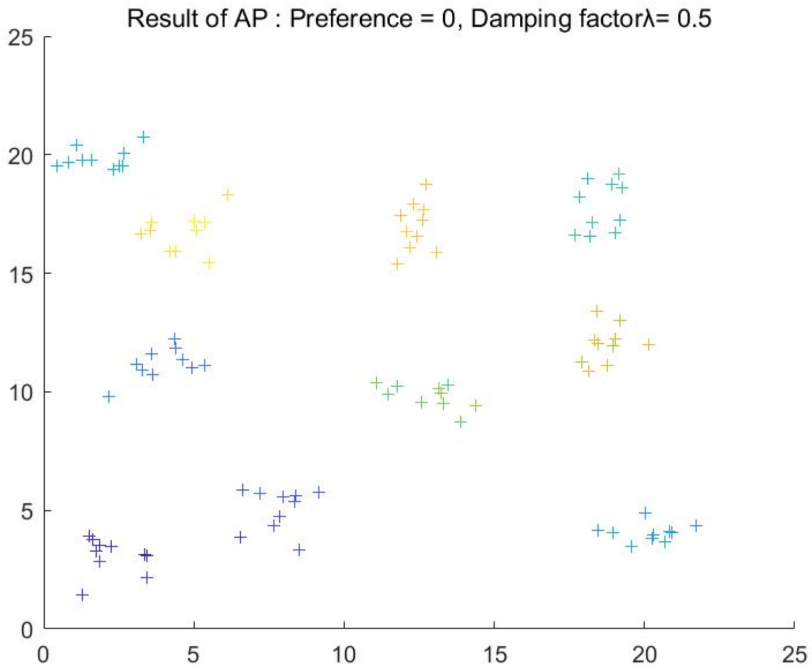
(b) AP's average squared error figures do not change but k-means graphics continuously fluctuate.
Affinity propagation is able to avoid many of the poor solutions caused by unlucky initialization.

(c) It does not converge. This is likely because damping factor is very low. A low damping factor causes large updates toward a minimum which can cause AP to oscillate. To avoid oscillations, we could increase the damping factor.

(d) Low preferences lead to small numbers of clusters and high preferences lead to large numbers of clusters. If preference is set to 0, every point will be an exemplar for itself.


Result of AP : Preference = -229.734, Damping factorλ= 0.5
0
5
10
15
20
25
0
5
10
15
20
25


Result of AP : Preference = -11.4867, Damping factorλ= 0.5
0
5
10
15
20
25
0
5
10
15
20
25


Result of AP : Preference = 0, Damping factorλ= 0.5
25
20
15
10
5
0
0
5
10
15
20
25