Clustering, K-Means, EM Tutorial

Kamyar Ghasemipour

Parts taken from Shikhar Sharma, Wenjie Luo, and Boris Ivanovic's tutorial slides, as well as lecture notes

Organization:

- Clustering
  - Motivation
- K-Means
  - Review & Demo
- Gaussian Mixture Models
  - Review
- EM Algorithm (time permitting)
  - Free Energy Justification

Clustering

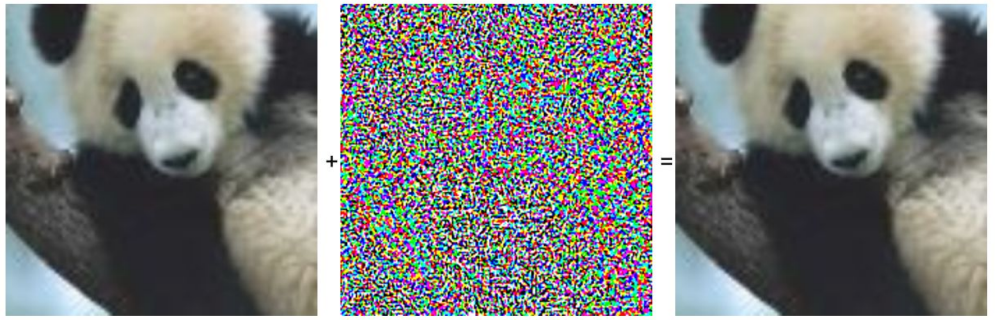
Clustering: Motivation

- Important assumption we make when doing any form of learning:

  “Similar data-points have similar behaviour”

- Eg. In the context of supervised learning

  “Similar inputs should lead to similar predictions”*



Original image classified as a panda with 60% confidence.

Tiny adversarial perturbation.

Imperceptibly modified image, classified as a gibbon with 99% confidence.

*sometimes our trained models don’t follow these assumption (cf. literature on adversarial examples)

Clustering: Examples

- Discretizing colours for compression using a codebook

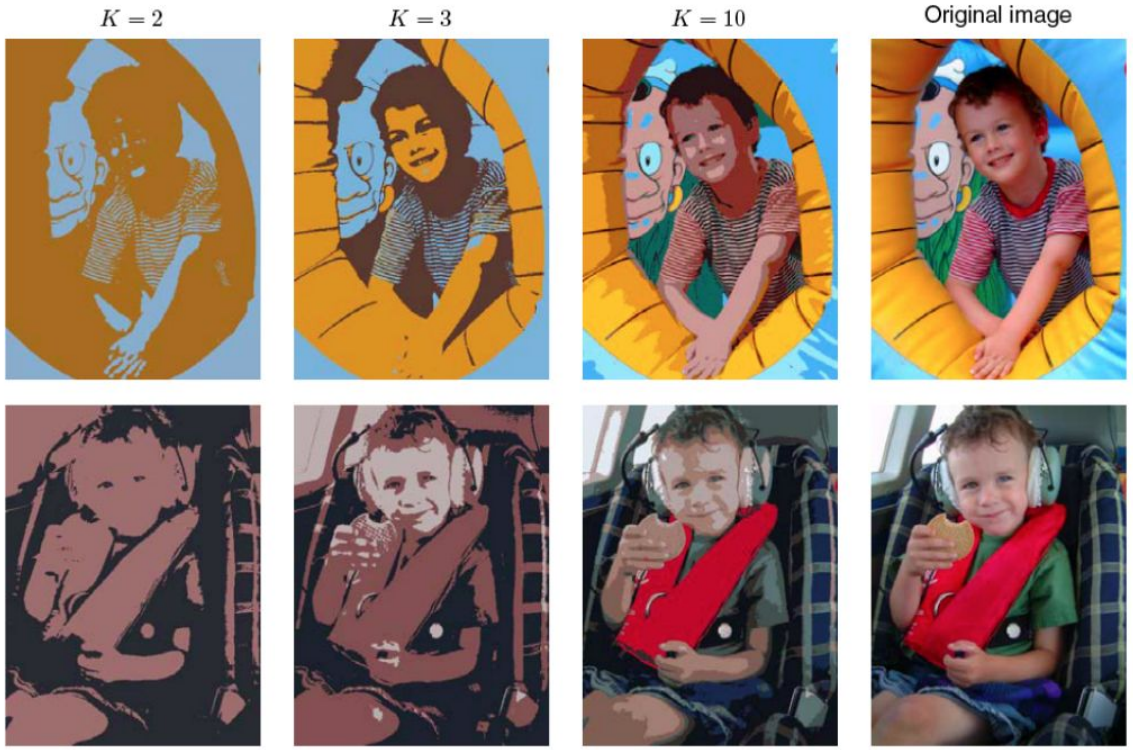
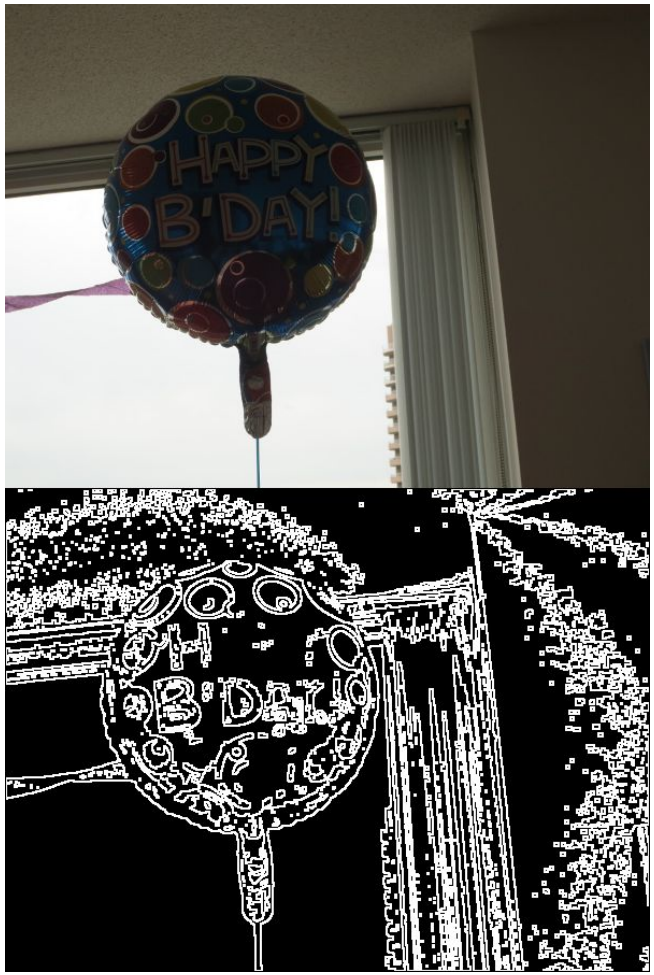


Figure from Bishop

Clustering: Examples

- Doing a very basic form of boundary detection
  - Discretize colours
  - Draw boundaries between colour groups



Clustering: Examples

- Like all unsupervised learning algorithms, clustering can be incorporated into the pipeline for training a supervised model
- We will go over an example of this very soon

Clustering: Challenges

- What is a good notion of “similarity”?
- Euclidean distance bad for Image

Clustering: Challenges

- The notion of similarity used can make the same algorithm behave in very different ways and can in some cases be a motivation for developing new algorithms (not necessarily just for clustering algorithms)
- Another question is how to compare different clustering algorithms
  - May have specific methods for making these decisions based on the clustering algorithms used
  - Can also use performance on down-the-line tasks as a proxy when choosing between different setups

Clustering: Some Specific Algorithms

- Today we shall review:
  - K-Means
  - Gaussian Mixture Models
- Hopefully there will be some time to go over EM as well

K-Means

K-Means: The Algorithm

1. Initialize K centroids
2. Iterate until convergence
   a. Assign each data-point to it's closest centroid
   b. Move each centroid to the center of data-points assigned to it

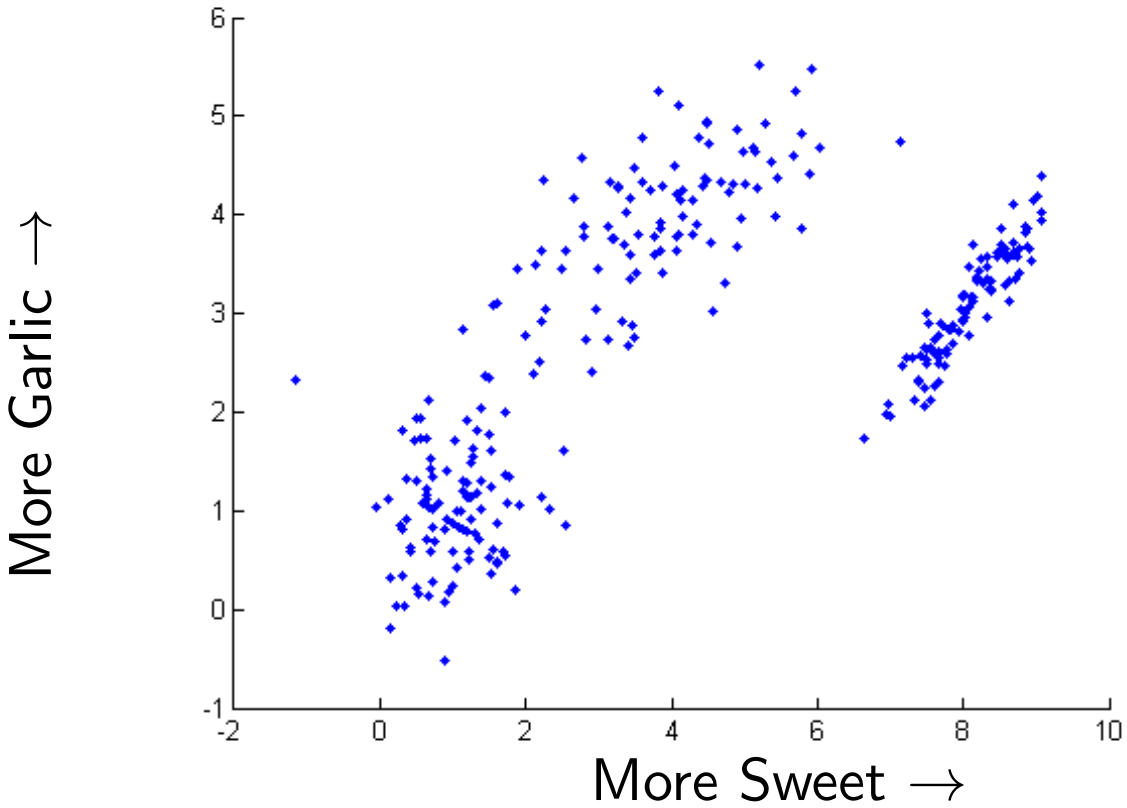
K-Means: A look at how it can be used

<< Slides from TA's past >>

Tomato sauce

- A major tomato sauce company wants to tailor their brands to sauces to suit their customers
- They run a market survey where the test subject rates different sauces
- After some processing they get the following data
- Each point represents the preferred sauce characteristics of a specific person

Tomato sauce data



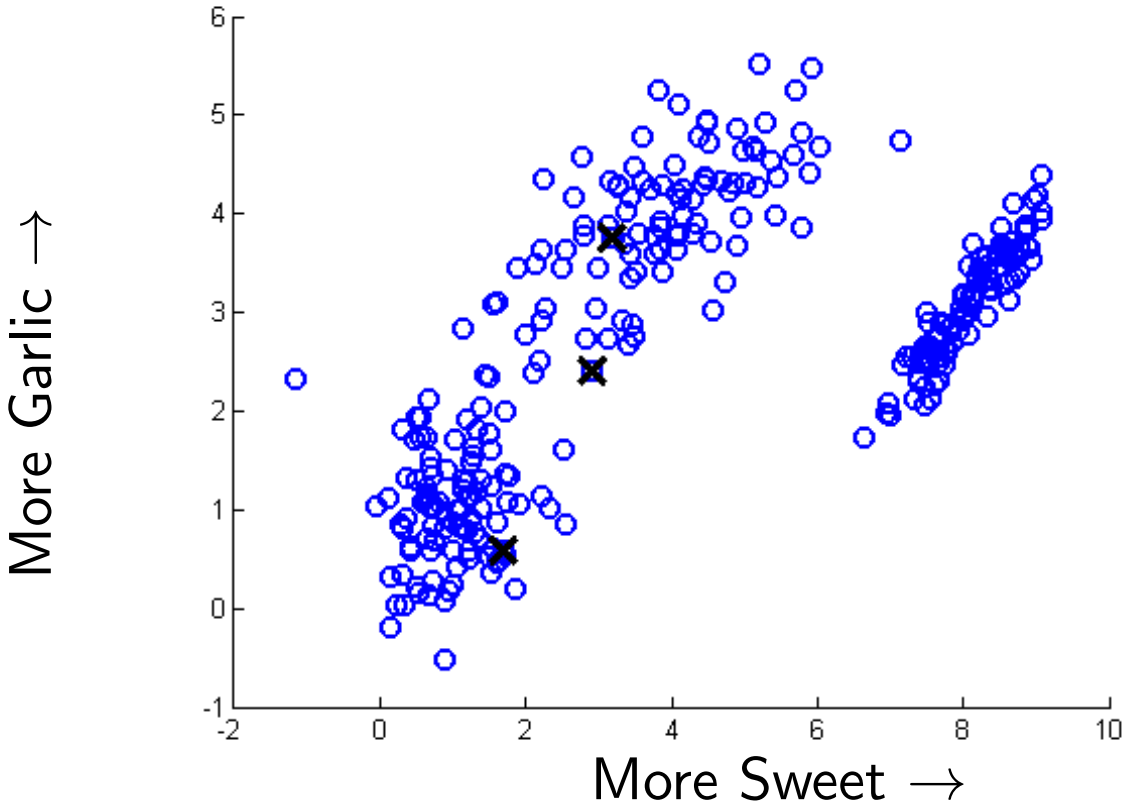
This tells us how much different customers like different flavors

Some natural questions

- How many different sauces should the company make?
- How sweet/garlicy should these sauces be?
- Idea: We will segment the consumers into groups (in this case 3), we will then find the best sauce for each group

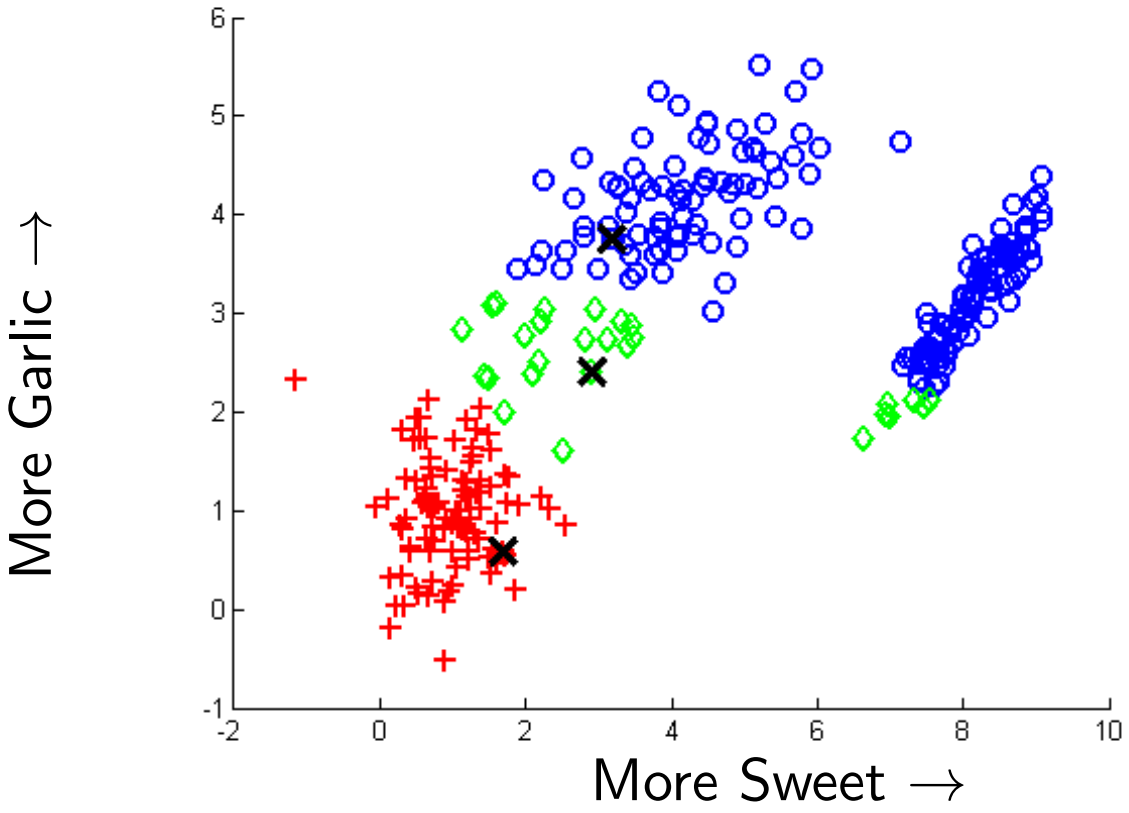
Approaching k-means

- Say I give you 3 sauces whose garlicy-ness and sweetness are marked by X



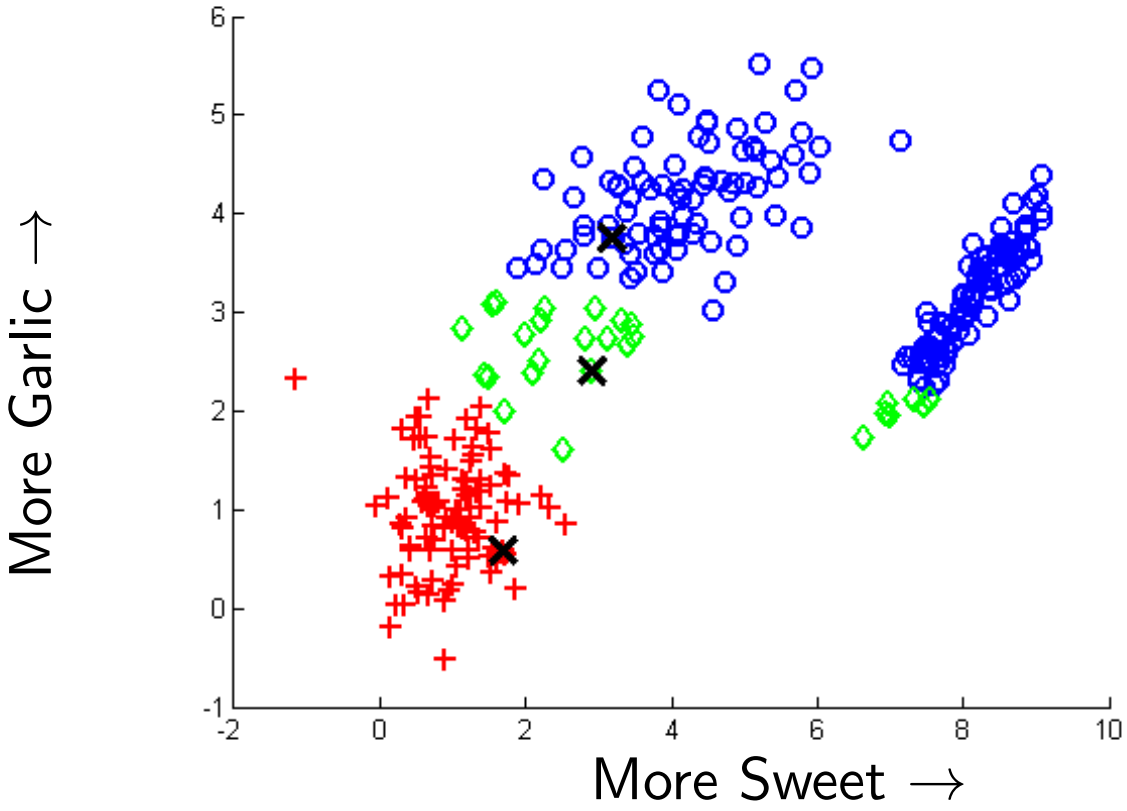
Approaching k-means

- We will group each customer by the sauce that most closely matches their taste



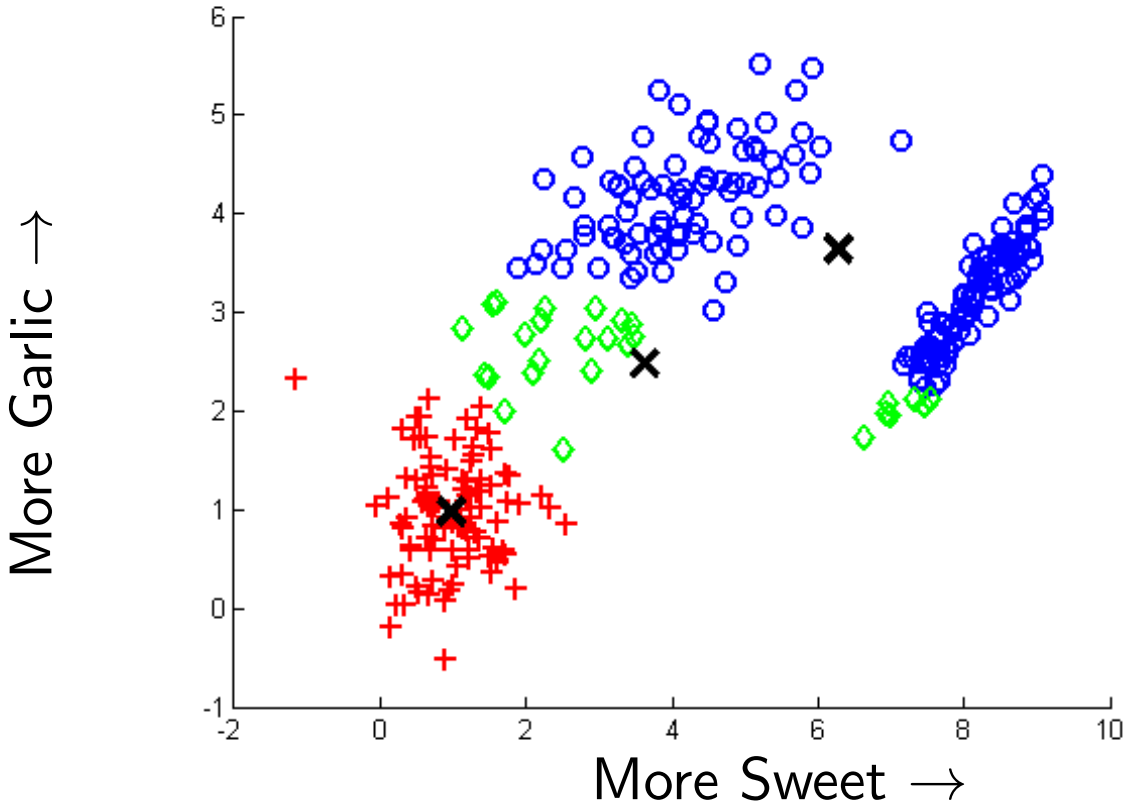
Approaching k-means

- Given this grouping, can we choose sauces that would make each group happier on average?



Approaching k-means

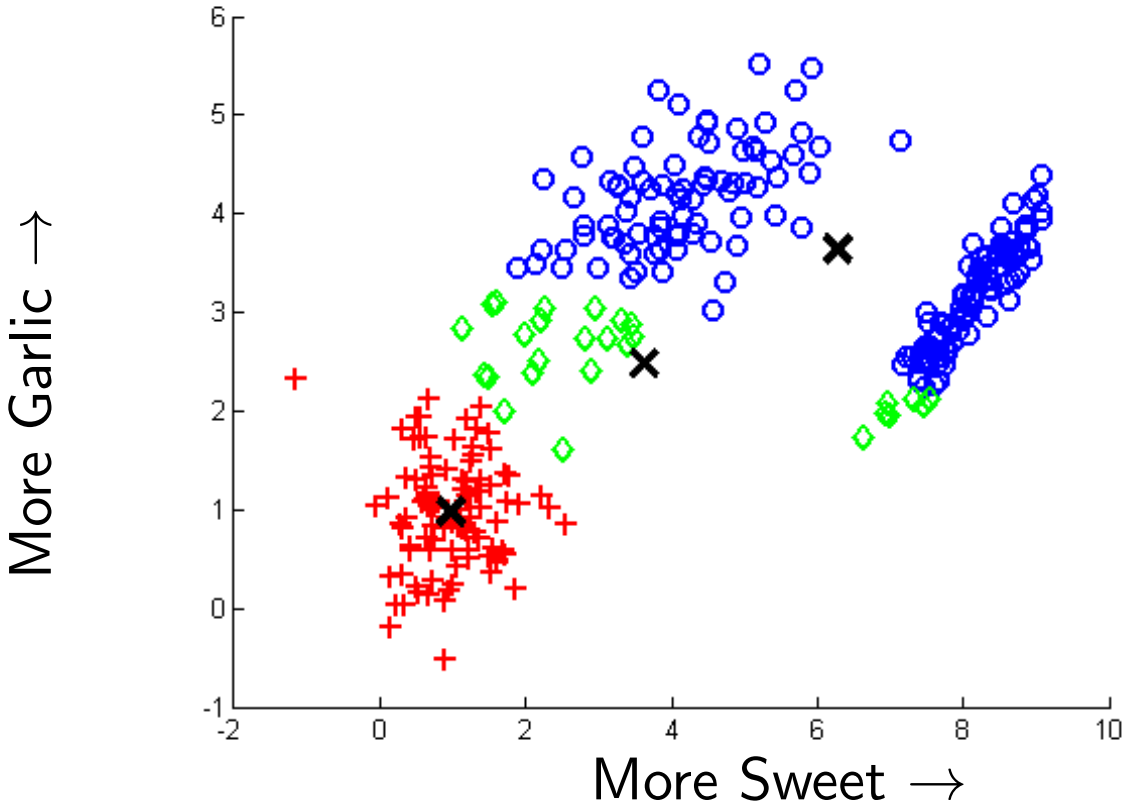
- Given this grouping, can we choose sauces that would make each group happier on average?



Yes !

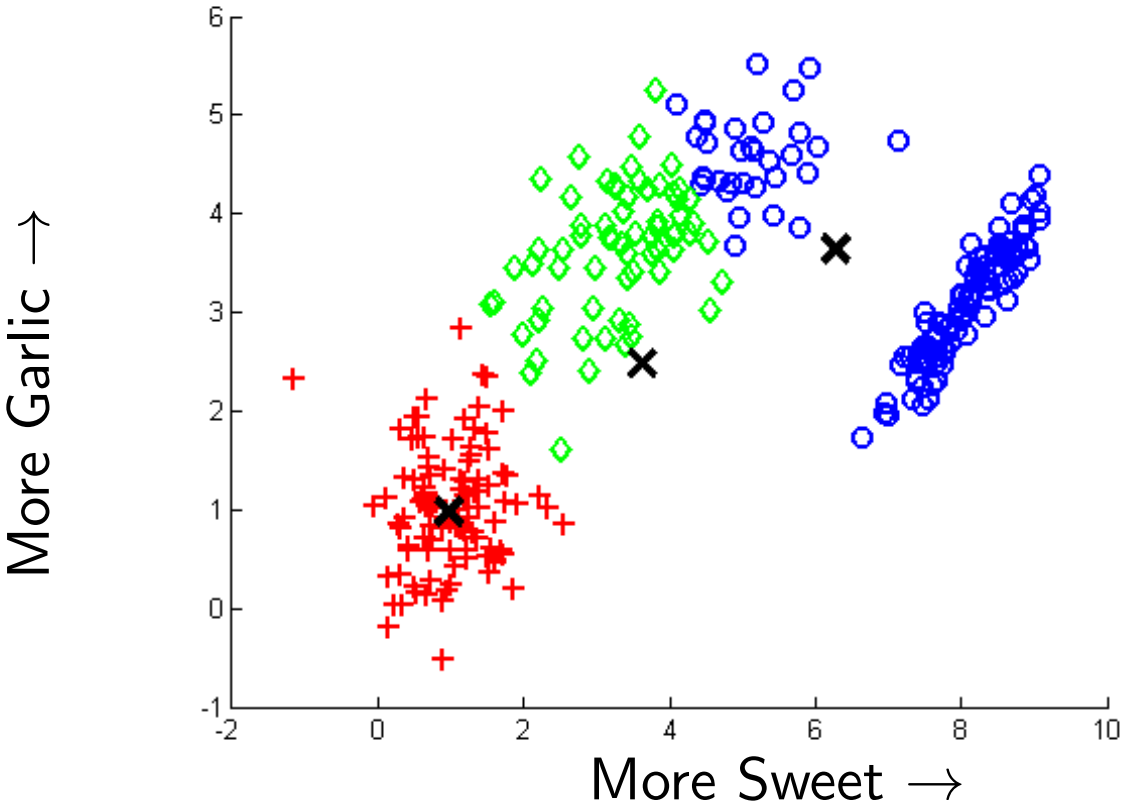
Approaching k-means

- Given these new sauces, we can regroup the customers



Approaching k-means

- Given these new sauces, we can regroup the customers



K-Means: Challenges

- How to initialize?
  - You saw k-means++ in lecture slides
  - Can come up with other heuristics
- How do you choose K?
  - You may come up with criteria for the value of K based on:
    - Restrictions on the magnitude of K
      - Everyone can’t have their own tomato sauce
    - Performance on some down-the-line task
      - If used for doing supervised learning later, must choose K such that you do not under/over fit

K-Means: Challenges

- K-Means algorithm converges to a local minimum:
  - Can try multiple random restarts
  - Other heuristics such as splitting discussed in lecture

- **Questions about K-Means?**

Gaussian Mixture Models

Generative Models

- One important class of methods in machine learning
- The goal is to define some parametric family of probability distributions and then maximize the likelihood of your data under this distribution by finding the best parameters

Back to GMM

- A Gaussian mixture distribution:

$$p(\mathbf{x}) = \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}|\mu_k, \Sigma_k)$$

- We had: $z \sim \text{Categorical}(\boldsymbol{\pi})$ (where $\pi_k \geq 0, \quad \sum_k \pi_k = 1$)
- Joint distribution: $p(\mathbf{x}, \mathbf{z}) = p(\mathbf{z})p(\mathbf{x}|\mathbf{z})$
- Log-likelihood:

$$\ell(\boldsymbol{\pi}, \mu, \Sigma) = \ln p(\mathbf{X}|\pi, \mu, \Sigma) = \sum_{n=1}^{N} \ln p(\mathbf{x}^{(n)}|\pi, \mu, \Sigma)$$

$$= \sum_{n=1}^{N} \ln \sum_{z^{(n)}=1}^{K} p(\mathbf{x}^{(n)}|\, z^{(n)}; \mu, \Sigma) p(z^{(n)}|\, \boldsymbol{\pi})$$

- Note: We have a hidden variable $z^{(n)}$ for every observation
- General problem: sum inside the log
- How can we optimize this?

Expectation Maximization for GMM Overview

- Elegant and powerful method for finding maximum likelihood solutions for models with latent variables
  1. E-step:
     - In order to adjust the parameters, we must first solve the inference problem: Which Gaussian generated each datapoint?
     - We cannot be sure, so it's a distribution over all possibilities.

$$\gamma_k^{(n)} = p(z^{(n)} = k|\mathbf{x}^{(n)}; \pi, \mu, \Sigma)$$

  2. M-step:
     - Each Gaussian gets a certain amount of posterior probability for each datapoint.
     - We fit each Gaussian to the weighted datapoints
     - We can derive closed form updates for all parameters

Gaussian Mixture Models: Connection to K-Means

- You saw soft K-means in lecture
- If you look at the update equations (and maybe some back of the envelope calculations) you will see that the update rule for soft k-means is the same as the GMMs where each Gaussian is spherical (0 mean, Identity covariance matrix)

Gaussian Mixture Models: Miscellany

- Can try initializing the centers with the k-means algorithm
- Your models will train a lot fast if you use diagonal covariance matrices (but it might not necessarily be a good idea)

EM

EM: Review

- The update rule for GMMs is a special case of the EM algorithm

EM: Free Energy Justification

- Let’s try doing this on the board

General EM Algorithm

1. Initialize Θ^{old}
2. E-step: Evaluate $p(\mathbf{Z}|\mathbf{X}, \Theta^{old})$ and compute

$$Q(\Theta, \Theta^{old}) = \sum_{z} p(\mathbf{Z}|\mathbf{X}, \Theta^{old}) \ln p(\mathbf{X}, \mathbf{Z}|\Theta)$$

3. M-step: Maximize

$$\Theta^{new} = arg \max_{\Theta} Q(\Theta, \Theta^{old})$$

4. Evaluate log likelihood and check for convergence (or the parameters). If not converged, $\Theta^{old} = \Theta^{new}$, Go to step 2

EM alternative approach *

- Our goal is to maximize

$$p(\mathbf{X}|\Theta) = \sum_{\mathbf{z}} p(\mathbf{X}, \mathbf{z}|\Theta)$$

- Typically optimizing $p(\mathbf{X}|\Theta)$ is difficult, but $p(\mathbf{X}, \mathbf{Z}|\Theta)$ is easy
- Let $q(\mathbf{Z})$ be a distribution over the latent variables. For any distribution $q(\mathbf{Z})$ we have

$$\ln p(\mathbf{X}|\Theta) = \mathcal{L}(q, \Theta) + KL(q||p(\mathbf{Z}|\mathbf{X}, \Theta))$$

where

$$\begin{aligned} \mathcal{L}(q, \Theta) &= \sum_{\mathbf{Z}} q(\mathbf{Z}) \ln \left\{ \frac{p(\mathbf{X}, \mathbf{Z}|\Theta)}{q(\mathbf{Z})} \right\} \\ KL(q||p) &= -\sum_{\mathbf{Z}} q(\mathbf{Z}) \ln \left\{ \frac{p(\mathbf{Z}|\mathbf{X}, \Theta)}{q(\mathbf{Z})} \right\} \end{aligned}$$

E-step and M-step *

$$\ln p(\mathbf{X}|\Theta) = \mathcal{L}(q, \Theta) + KL(q||p(\mathbf{Z}|\mathbf{X}, \Theta))$$

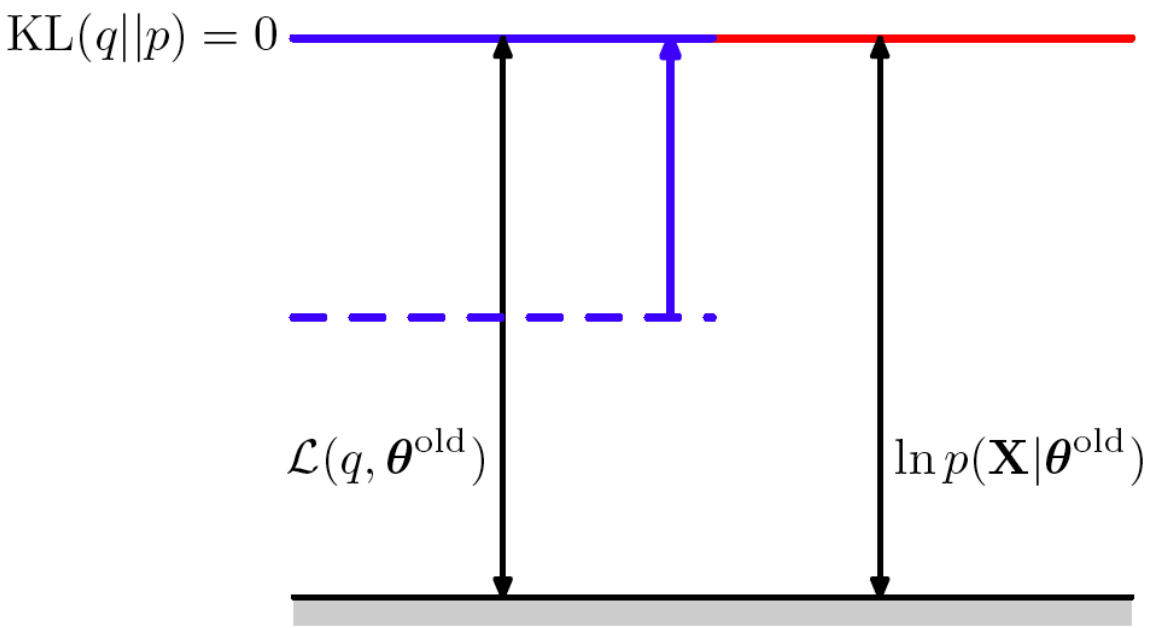
- In the E-step we maximize $q(\mathbf{Z})$ w.r.t the lower bound $\mathcal{L}(q, \Theta^{old})$
- This is achieved when $q(\mathbf{Z}) = p(\mathbf{Z}|\mathbf{X}, \Theta^{old})$
- The lower bound $\mathcal{L}$ is then

$$\begin{aligned}\mathcal{L}(q, \Theta) &= \sum_{\mathbf{Z}} p(\mathbf{Z}|\mathbf{X}, \Theta^{old}) \ln p(\mathbf{X}, \mathbf{Z}|\Theta) - \sum_{\mathbf{Z}} p(\mathbf{Z}|\mathbf{X}, \Theta^{old}) \ln p(\mathbf{Z}|\mathbf{X}, \Theta^{old}) \\ &= Q(\Theta, \Theta^{old}) + \text{const}\end{aligned}$$

  with the content the entropy of the q distribution, which is independent of Θ

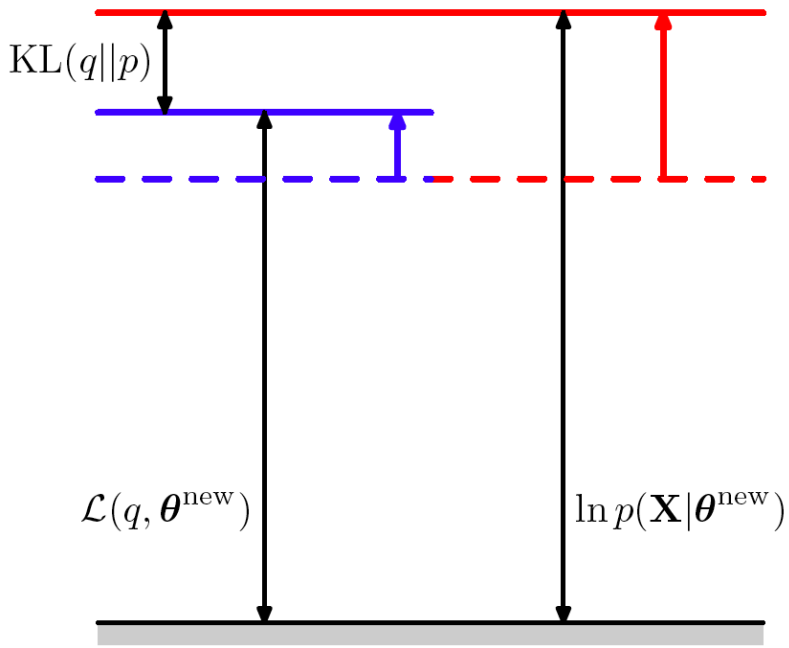
- In the M-step the quantity to be maximized is the expectation of the complete data log-likelihood
- Note that Θ is only inside the logarithm and optimizing the complete data likelihood is easier

Visualization of E-step



- The q distribution equal to the posterior distribution for the current parameter values Θ^{old}, causing the lower bound to move up to the same value as the log likelihood function, with the KL divergence vanishing.

Visualization of M-step



- The distribution $q(\mathbf{Z})$ is held fixed and the lower bound $\mathcal{L}(q, \Theta)$ is maximized with respect to the parameter vector Θ to give a revised value Θ^{new}. Because the KL divergence is nonnegative, this causes the log likelihood $\ln p(\mathbf{X}|\Theta)$ to increase by at least as much as the lower bound does.