CSC411 Tutorial #6
Clustering: K-Means, GMM, EM

March 11, 2016

Boris Ivanovic*

csc411ta@cs.toronto.edu

*Based on the tutorial by Shikhar Sharma and Wenjie Luo's 2014 slides.

Outline for Today

- K-Means
- GMM
- Questions

- I’ll be focusing more on the intuitions behind these models, the math is not as important for your learning here

Clustering

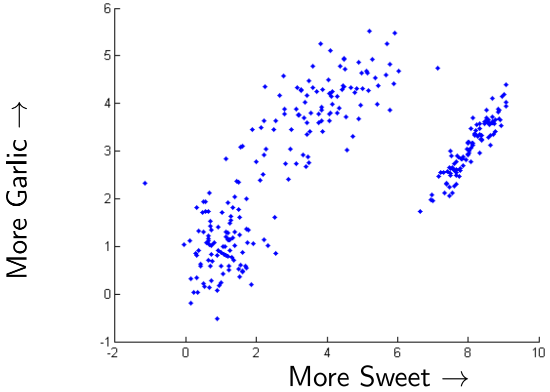
- In classification, we are given data with associated labels
- What if we aren't given any labels? Our data might still have structure
- We basically want to simultaneously label points and build a classifier

PS. I didn't change the bottom information because that would be disingenuous of me, and also because credit should be given where credit is due. Thanks Shikhar for the tutorial slides!

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



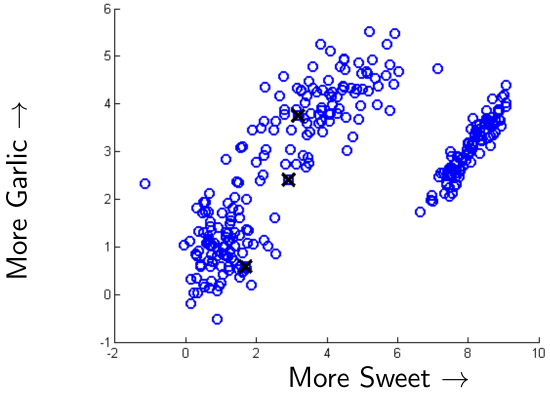
This tells us how much different customers like different flavors

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

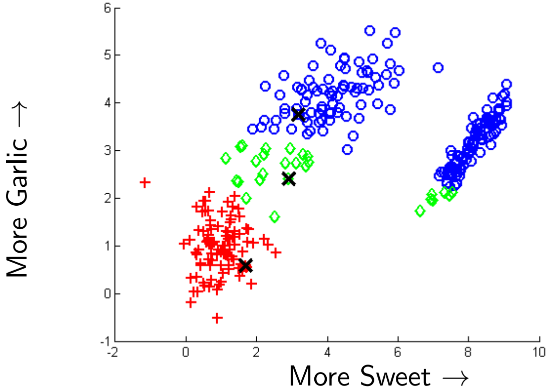
Approaching k-means

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



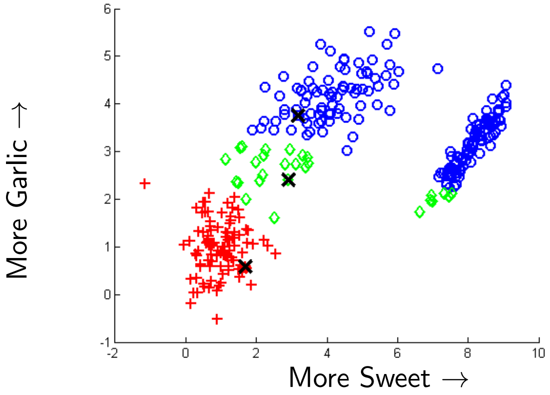
Approaching k-means

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



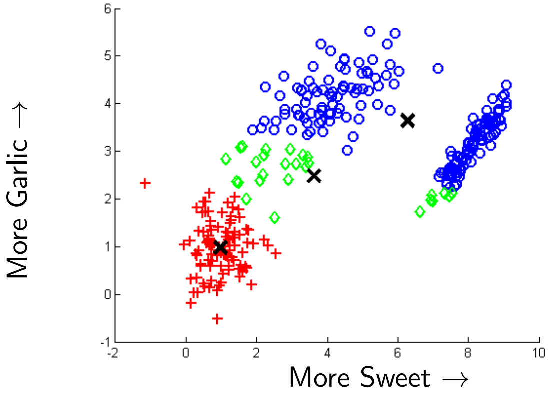
Approaching k-means

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



Approaching k-means

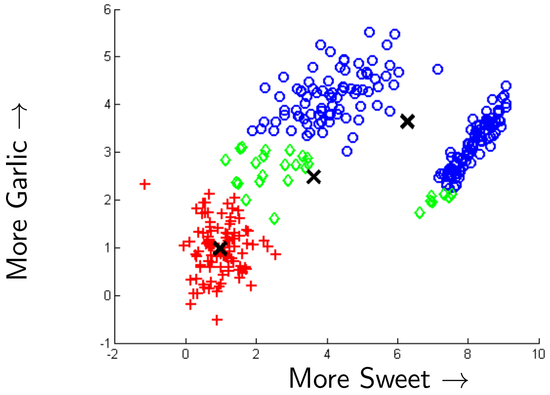
- Given this grouping, can we choose sauces that would make each group happier on average?



Yes !

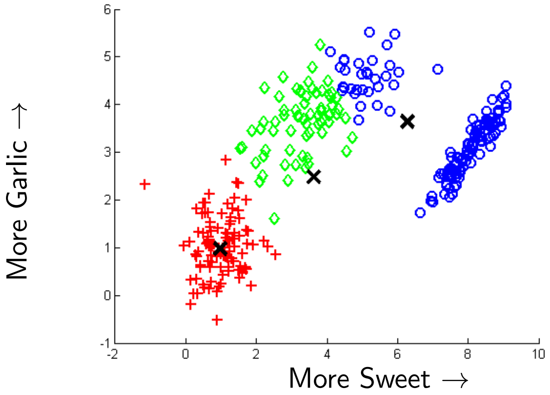
Approaching k-means

- Given these new sauces, we can regroup the customers



Approaching k-means

- Given these new sauces, we can regroup the customers



The k-means algorithm

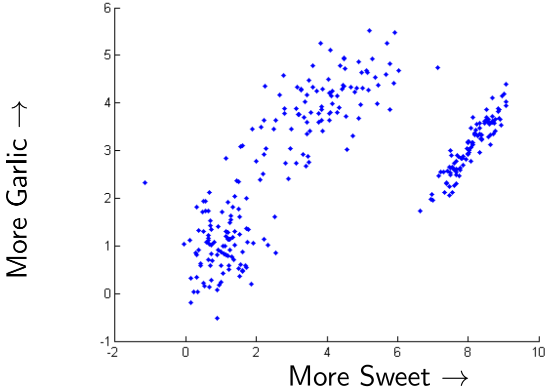
- **Initialization:** Choose k random points to act as cluster centers
- Iterate until convergence:
  - **Step 1:** Assign points to closest center (forming k groups)
  - **Step 2:** Reset the centers to be the mean of the points in their respective groups

Viewing k-means in action

- **Demo...**
- **Note:** K-Means only finds a local optimum
- Questions:
  - How do we choose k?
    - Couldn't we just let each person have their own sauce? (Probably not feasible...)
  - Can we change the distance measure?
    - Right now we're using Euclidean
  - Why even bother with this when we can "see" the groups? (Can we plot high-dimensional data?)

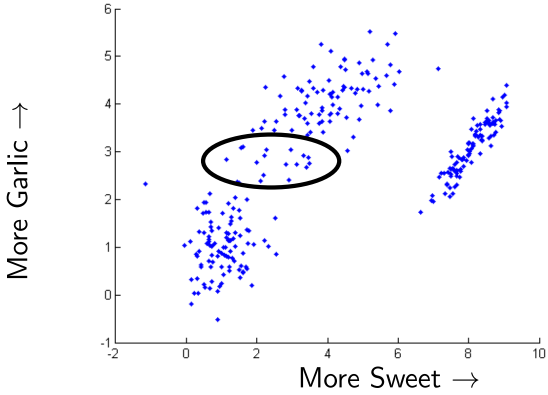
A "simple" extension

- Let's look at the data again, notice how the groups aren't necessarily circular?



A "simple" extension

- Also, does it make sense to say that points in this region belong to one group or the other?



Flaws of k-means

- It can be shown that k-means assumes the data belong to spherical groups, moreover it doesn't take into account the variance of the groups (size of the circles)
- It also makes hard assignments, which may not be ideal for ambiguous points
    - This is especially a problem if groups overlap
- We will look at one way to correct these issues

Isotropic Gaussian mixture models

- K-means implicitly assumes each cluster is an isotropic (spherical) Gaussian, it simply tries to find the optimal mean for each Gaussian
- However, it makes an additional assumption that each point belongs to a single group
- We will correct this problem first by allowing each point to "belong to multiple groups"
  - More accurately, that it belongs to each group with probability p_i, where $\sum_i p_i = 1$

Gaussian mixture models

- Given a data point x with dimension D:
- A multivariate isotropic Gaussian PDF is given by:

$$P(x) = (2\pi)^{-\frac{D}{2}}(\sigma^2)^{-\frac{D}{2}} e^{-\frac{1}{2\sigma^2}(x-\mu)^T(x-\mu)} \tag{1}$$

- A multivariate Gaussian in general is given by:

$$P(x) = (2\pi)^{-\frac{D}{2}}|\Sigma|^{-\frac{1}{2}} e^{-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)} \tag{2}$$

- We can try to model the covariance as well to account for elliptical clusters

Gaussian mixture models

- Demo GMM with full covariance
- Notice that now it takes much longer to converge
- Can be much faster convergence by first initializing with k-meansThe EM algorithm

THE EM algorithm

- What we have just seen is an instance of the EM algorithm
- The EM algorithm is actually a meta-algorithm, it tells you the steps needed in order to derive an algorithm to learn a model
- The "E" stands for expectation, the "M" stands for maximization
- We will look more closely at what this algorithm does, but won't go into extreme detail

EM for the Gaussian Mixture Model

- Recall that we are trying to put the data into groups, while simultaneously learning the parameters of that group
- If we knew the groupings in advance, the problem would be easy
    - With k groups, we are just fitting k separate Gaussians
    - With soft assignments, the data is simply weighted (i.e. we calculate weighted means and covariances)

EM for the Gaussian Mixture Model

- Given initial parameters:
- Iterate until convergence
  - E-step:
    - Partition the data into different groups (soft assignments)
  - M-step:
    - For each group, fit a Gaussian to the weighted data belonging to that group

EM in general

- We specify a model that has variables (x, z) with parameters θ, denote this by $P(x, z|\theta)$
- We want to optimize the log-likelihood of our data
  - $\log(P(x|\theta)) = \log(\sum_z P(x, z|\theta))$
- x is our data, z is some variable with extra information
  - Cluster assignments in the GMM, for example
- We don't know z, it is a "latent variable"
- E-step: infer the expected value for z given x
- M-step: maximize the "complete data log-likelihood" $\log(P(x, z|\theta))$ with respect to θ