



Unsupervised Clustering

Mark Dredze
Machine Learning
CS 601.475

Today

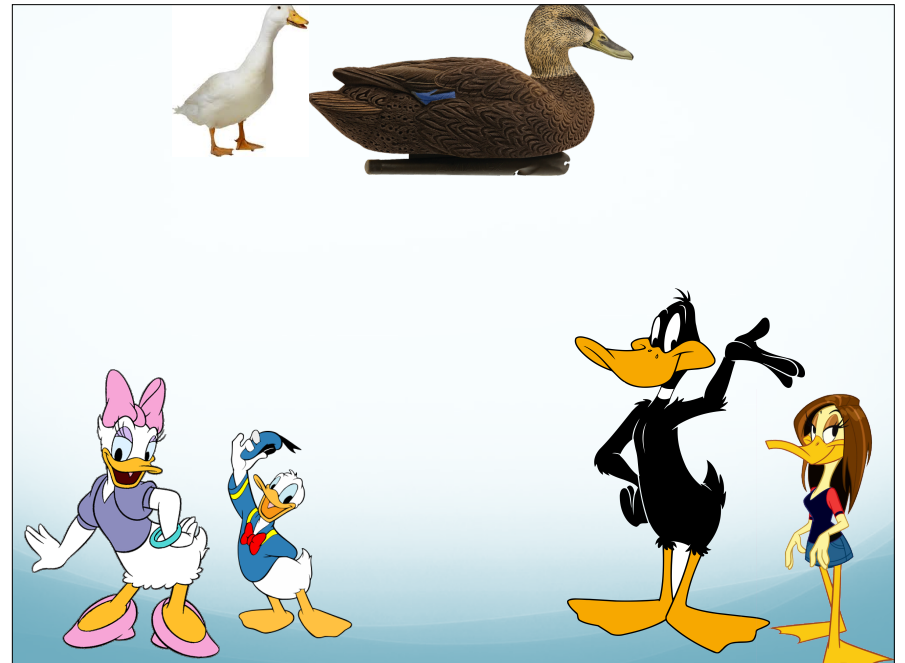
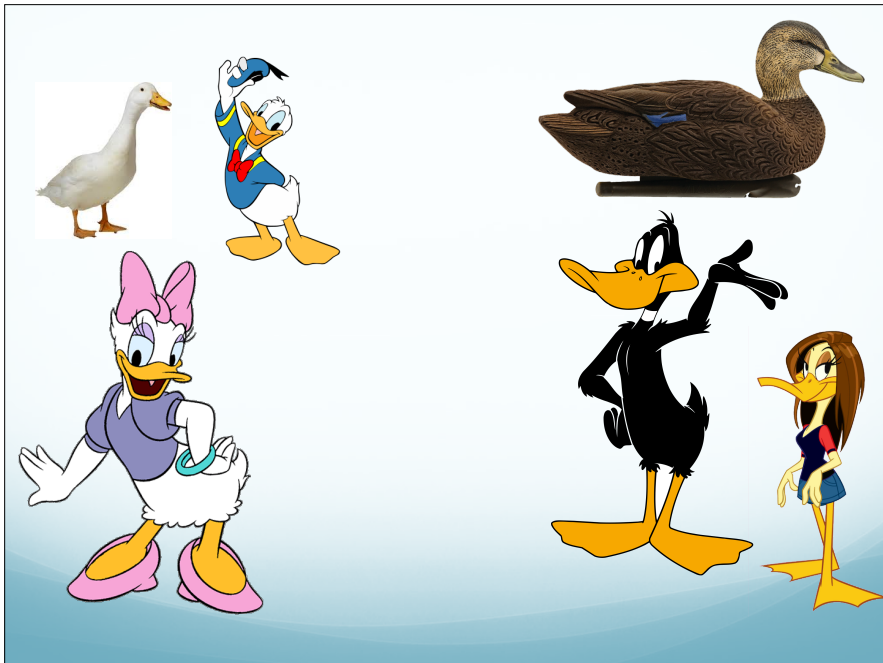
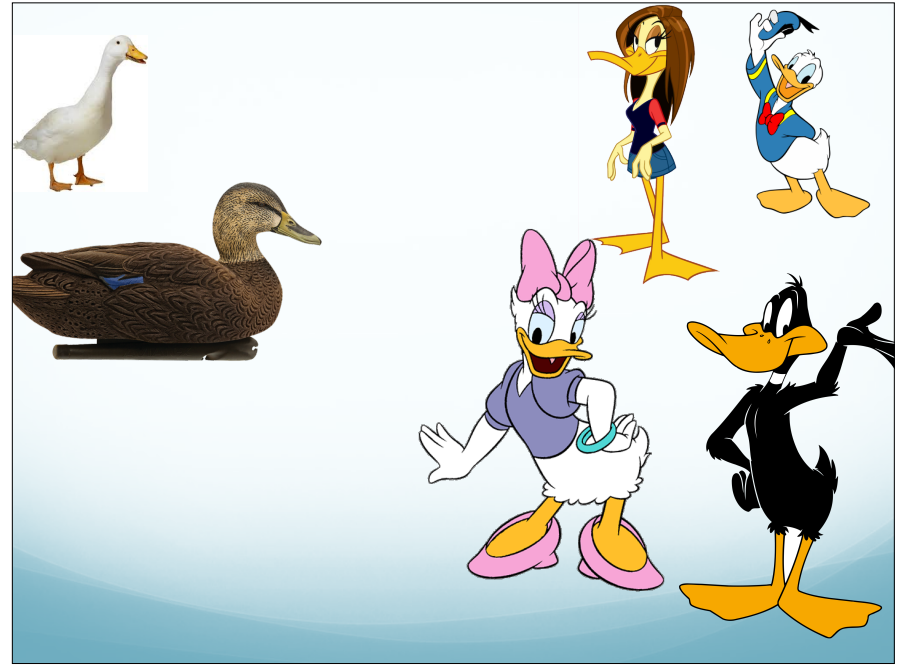
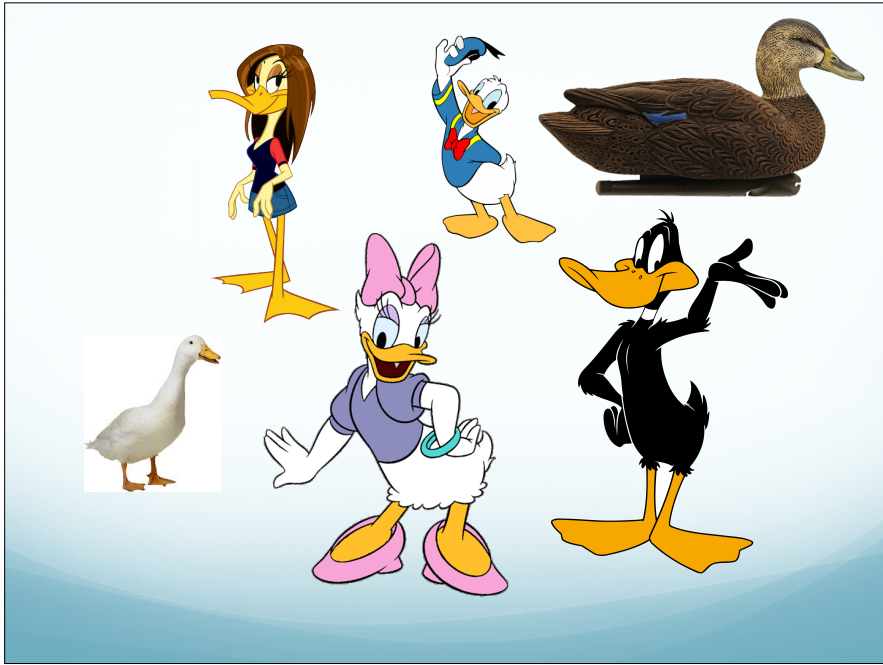
- We will focus on specific algorithms
- Start with discrete clustering
- Later transition to probabilistic methods

Unsupervised Learning

- No labels
 - Can't do classification anymore!
- We still can have a notion of **groups**
- Task: divide things into piles of similar things
- Classification found patterns that explained a label
 - We can find patterns that separate the data

Clustering

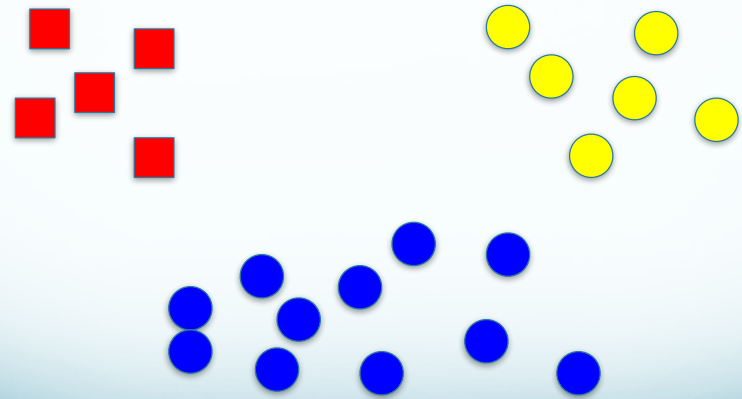
- Sort the data into clusters (groups)
- Examples that are in the same group are similar
 - Should be more similar to one another than to items not in the cluster
 - Ideally clusters correspond to (unknown) class labels
- We don't know what we will get (unsupervised!)
 - What does it mean for two examples to be similar?



Clustering

- Data $\{(\mathbf{x}_i)\}_{i=1}^N$ $\mathbf{x}_i \in \mathbb{R}^M$
- Input: number of clusters k
- Algorithm: partition data into k clusters
 - Each \mathbf{x} belongs to a cluster
- Cluster: a group of similar examples

Geometric Model



Solving Clustering

- How do we group examples into clusters?
- Same as before!
 - Design a model
 - Define a model objective to represent learning goal
 - Write procedure for maximizing objective
 - Compute model parameters using procedure

Defining Clusters

- A cluster is a group of similar examples
- Define cluster k by a prototype μ_k
- $r_{nk} \in \{0,1\}$, value of 1 means example n in cluster k
- Mean of the examples in cluster k

$$\mu_k = \frac{1}{\sum_{n=1}^N r_{nk}} \sum_{n=1}^N r_{nk} \mathbf{x}_n$$

Objective

- What are good clusters?
 - A group of points that is maximally similar
- Objective: maximize the similarity of every cluster
- Distortion measure

$$J = \sum_{n=1}^N \sum_{k=1}^K r_{nk} \|\mathbf{x}_n - \boldsymbol{\mu}_k\|^2$$

- Each example measured as distance from prototype
 - Euclidean distance
- This is similar to sum of squares error

Learning

$$J = \sum_{n=1}^N \sum_{k=1}^K r_{nk} \|\mathbf{x}_n - \boldsymbol{\mu}_k\|^2$$

- Notice there are two parameters: $\boldsymbol{\mu}_k$ and r_{nk}
- Learning: select values to minimize J

Learning

- Note that $\boldsymbol{\mu}_k$ and r_{nk} are dependent on each other
 - If we knew $\boldsymbol{\mu}_k$ we could set r_{nk}
 - Assign each point to closest cluster
 - If we knew r_{nk} we could set $\boldsymbol{\mu}_k$
 - Compute cluster means from examples in cluster
- Strategy: iterative procedure
 - Select $\boldsymbol{\mu}_k$ that minimizes J with fixed r_{nk}
 - Select r_{nk} that minimizes J with fixed $\boldsymbol{\mu}_k$

Update Rules

- Take the derivative of J with respect to each parameter
 - Set to 0 and solve for the parameter

$$r_{nk} = \begin{cases} 1 & \text{if } k = \arg \min_j \|\mathbf{x}_n - \boldsymbol{\mu}_j\|^2 \\ 0 & \text{otherwise} \end{cases}$$

$$\boldsymbol{\mu}_k = \frac{\sum_n r_{nk} \mathbf{x}_n}{\sum_n r_{nk}}$$

Convergence

- Each update reduces the value of J
 - Therefore it will converge
- Note: J is non-convex
 - Resulting value may not be the best
 - Initial values matter!

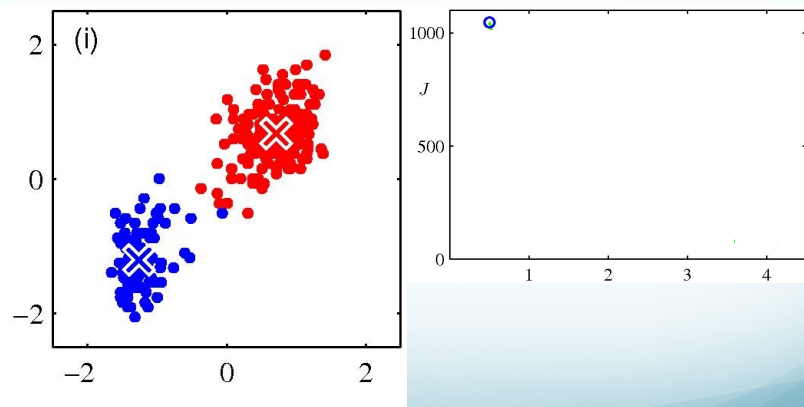
Algorithm: K-Means

- Given data $\{(\mathbf{x}_i)\}_{i=1}^N$ $\mathbf{x}_i \in \mathbb{R}^M$
- Initialize μ_k
- Iteratively update until convergence:

$$r_{nk} = \begin{cases} 1 & \text{if } k = \arg \min_j \|\mathbf{x}_n - \mu_j\|^2 \\ 0 & \text{otherwise} \end{cases}$$

$$\mu_k = \frac{\sum_n r_{nk} \mathbf{x}_n}{\sum_n r_{nk}}$$

Illustration

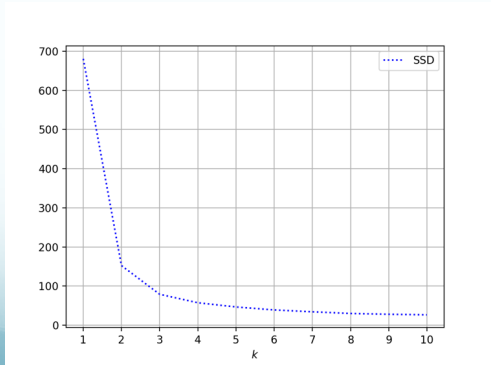


Differences from Classification

- No train and test data
 - We have no labels, so use all available data
- How do we choose k ?
 - Requires human input
- Evaluation measure
 - What do we compare against?
 - Compare J against other clustering results
 - Sometimes we have some labeled data for evaluation only

Choosing K

- Mean square distance vs k
- Look for the “elbow” in the plot as we increase k



Choosing K

- Use BIC or AIC
- Includes penalty for adding more clusters

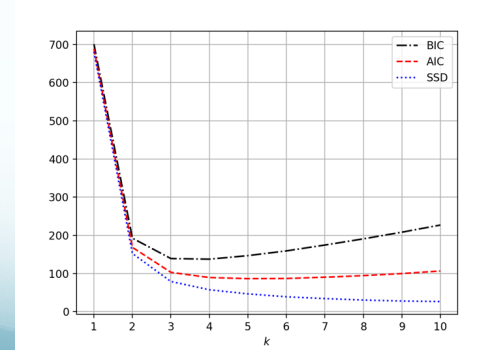


Image Compression and Segmentation

- Even this simple algorithm yields powerful results
- Image compression/segmentation
 - Each pixel is represented using RGB values
 - Cluster pixels using K-means
 - Note: ignores position of pixel in image

Image Segmentation



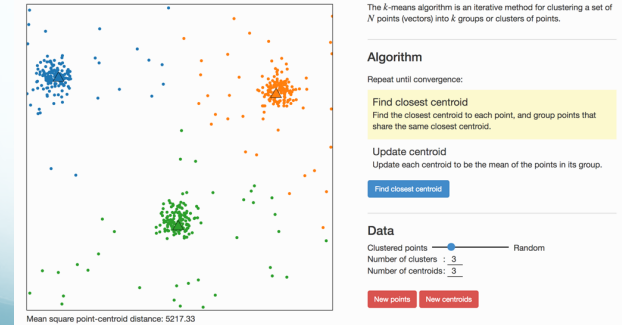
K-Means Issues

- Computational Complexity?
 - Re-assignment step:
 - Vector distance- M operations
 - Find best cluster for each example: $K*N$ distances
 - Total: $O(KNM)$
 - Compute new means:
 - Each example added to cluster once- $O(NM)$
 - For I iterations, total is $O(IKNM)$
 - Linear in each variable
 - Still slow compared to some supervised methods
- Difficulty of finding optimal assignment?
 - NP-Hard in Euclidean space (certainly non-convex)
 - Solution: Random restarts

K-Means Demo

<http://stanford.edu/class/ee103/visualizations/kmeans/kmeans.html>

Visualizing K-Means Clustering



Problems with K-Means

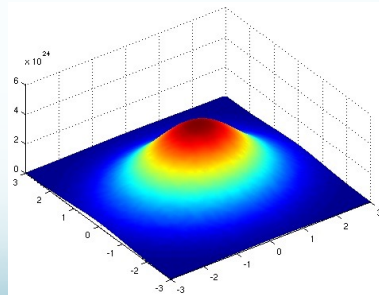
- Hard assignment
 - Examples go from one cluster to the other right away
- A smooth transition might be better
- How do we do smooth transitions?
 - Probabilities!

Generative Clustering Model

- Assume we have K clusters
- Each cluster represented by a multi-variate Gaussian
- Generative process:
 - Select a cluster (a Gaussian distribution)
 - Generate an example by sampling from the Gaussian

Gaussian Mixtures

- Since we have multiple Gaussians generating points, we call the model Gaussian Mixture Model
- Why Gaussians?
 - Captures intuition about clusters
 - Examples are more likely to be near center of cluster



Gaussian Mixture Model

- (Derivation on board)

Gaussian Mixture Model

- Cluster means, variances, coefficients
- Cluster Responsibilities

$$N_k = \sum \gamma(z_{nk})$$

$$\pi_k = \frac{N_k}{N} = \frac{N_k}{\sum_k N_k}$$

$$\mu_k = \frac{1}{N_k} \sum_{n=1}^N \gamma(z_{nk}) \mathbf{x}_n$$

$$\Sigma_k = \frac{1}{N_k} \sum_{n=1}^N \gamma(z_{nk}) (\mathbf{x}_n - \mu_k)(\mathbf{x}_n - \mu_k)^T$$

$$\gamma(z_{nk}) = \frac{\pi_k \mathcal{N}(\mathbf{x}_n | \mu_k, \Sigma_k)}{\sum_{j=1}^K \pi_j \mathcal{N}(\mathbf{x}_n | \mu_j, \Sigma_j)}$$

Algorithm: GMMs

- Given data $\{(\mathbf{x}_i)\}_{i=1}^N \quad \mathbf{x}_i \in \mathbb{R}^M$
- Initialize $\mu_k \quad \Sigma_k \quad \pi_k$
- Iteratively update until convergence:
 - $\gamma(z_k)$
 - $\mu_k \quad \Sigma_k \quad \pi_k$

GMM Demo

<https://www.youtube.com/watch?v=TLg-fvTfqno>
<https://www.youtube.com/watch?v=uUtpiK5NEAM>

Problems with GMMs


- Mode collapse
 - When a single data point is isolated and becomes its own Gaussian
 - Watch for this and reset the Gaussian
- Very non-convex likelihood
 - K! equivalent best solutions
 - Can find any one, but may end up in a local minimum
- Requires more iterations to converge than K-Means
 - Plus, each iteration is more computationally expensive

Similarities

	K-Means	Gaussian Mixtures
Assign examples to clusters	r_{rk}	$\gamma(z_k)$
Compute new model parameters that maximize assignments	μ_k	$\mu_k \sum_k \pi_k$

Same Algorithm

- The maximization algorithm for both models is the same!
- Iterate two steps
 - Compute the **expected** cluster assignments according to the current model
 - **Maximize** the model parameters according to the current cluster assignments
- Expectation Maximization Algorithm (EM)



Next Time

The EM Algorithm