Lecture 12
Clustering, K-Means and EM

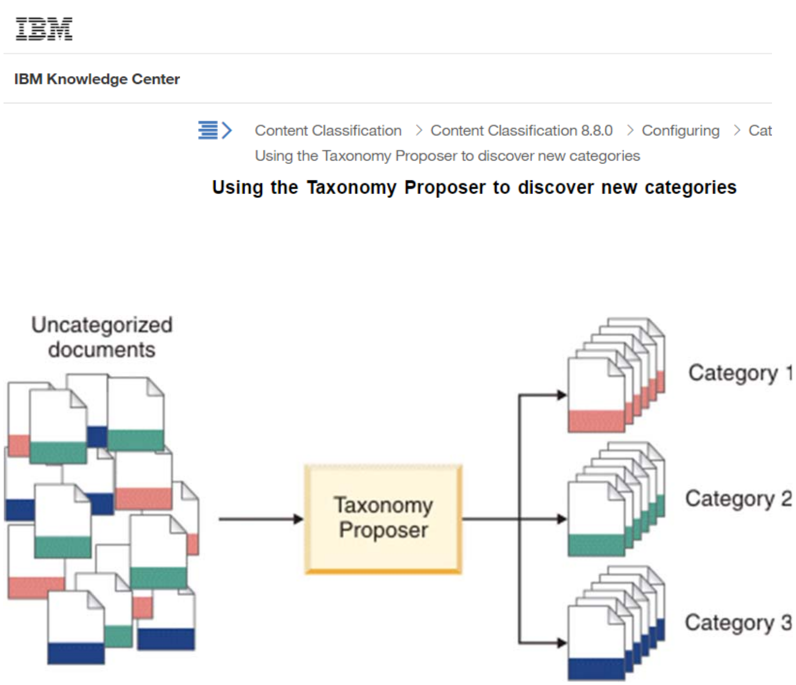
EE-UY 4563/EL-GY 9123: INTRODUCTION TO MACHINE LEARNING

PROF. SUNDEEP RANGAN

Outline

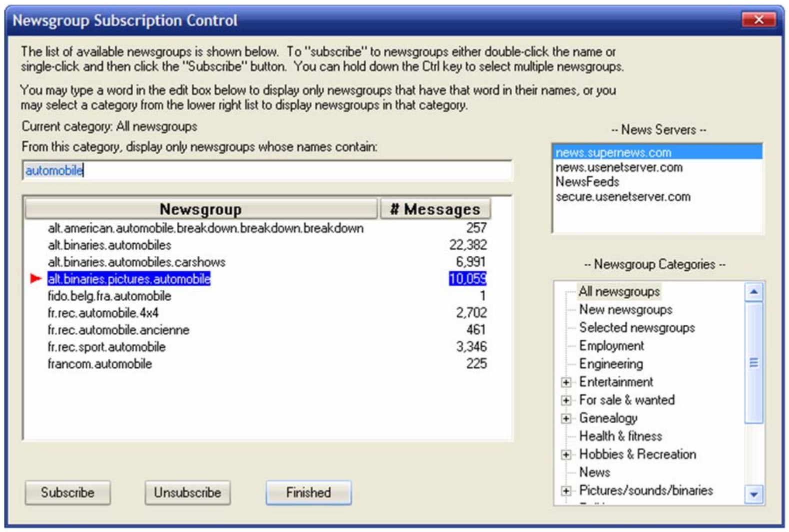
- Motivating Example: Document clustering
- K-means
- K-means for document clustering
- Latent semantic analysis
- Gaussian Mixture models (GMMs)
- Expectation Maximization (EM) fitting of GMMs
- Convergence of EM

Document Clustering



- ❑Data mining
- ❑Often have huge numbers of documents
- ❑How can we organize this?
- ❑Key idea: documents are often in clusters
- ❑Can we detect these clusters?
- ❑Can be a lucrative service
  - See IBM service to left

UseNet Newsgroups



- Began in late 1970s
- Discussion groups for various topics
  - Started on early university networks
  - Migrated to Internet
  - Peaked in 1990s
- Useful for studying clustering
  - Simple documents
  - “ground truth”: Docs have categories

Loading the Data

- See demo_doc_cluster.ipynb
- Taken from http://scikit-learn.org/stable/auto_examples/text/document_clustering.html
- Newsgroups built into sklearn

```
categories = [
    'alt.atheism',
    'talk.religion.misc',
    'comp.graphics',
    'sci.space',
]
# Uncomment the following to do the analysis on all the categories
#categories = None

print("Loading 20 newsgroups dataset for categories:")
print(categories)

dataset = fetch_20newsgroups(subset='all', categories=categories,
                             shuffle=True, random_state=42)
```

```
Loading 20 newsgroups dataset for categories:
['alt.atheism', 'talk.religion.misc', 'comp.graphics', 'sci.space']
```

A Typical Newsgroup Post

```
ind = 10
data_ex = dataset.data[ind]
cat_ex  = dataset.target_names[labels[ind]]
print('Post from {0:s}'.format(cat_ex))
print()
print(data_ex)
```

```
Post from comp.graphics

From: richter@fossi.hab-weimar.de (Axel Richter)
Subject: True Color Display in POV
Keywords: POV, Raytracing
Nntp-Posting-Host: fossi.hab-weimar.de
Organization: Hochschule fuer Architektur und Bauwesen Weimar, Germany
Lines: 6


Hallo POV-Renderers !
I've got a BocaX3 Card. Now I try to get POV displaying True Colors
while rendering. I've tried most of the options and UNIVESA-Driver
but what happens isn't correct.
Can anybody help me ?
```

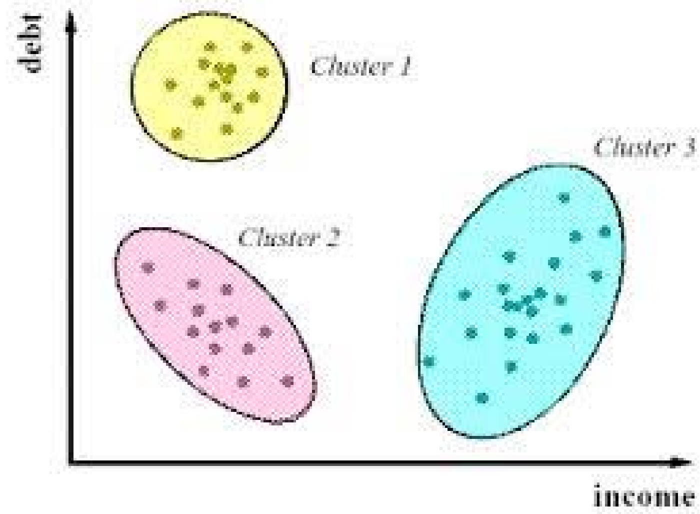
- Data for the posts are in:
  - Dataset.data
  - Dataset.labels
  - Dataset.target_names

Outline

- ❑Motivating Example: Document clustering
- ❑K-means
- ❑K-means for document clustering
- ❑Latent semantic analysis
- ❑Gaussian Mixture models (GMMs)
- ❑Expectation Maximization (EM) fitting of GMMs
- ❑Convergence of EM

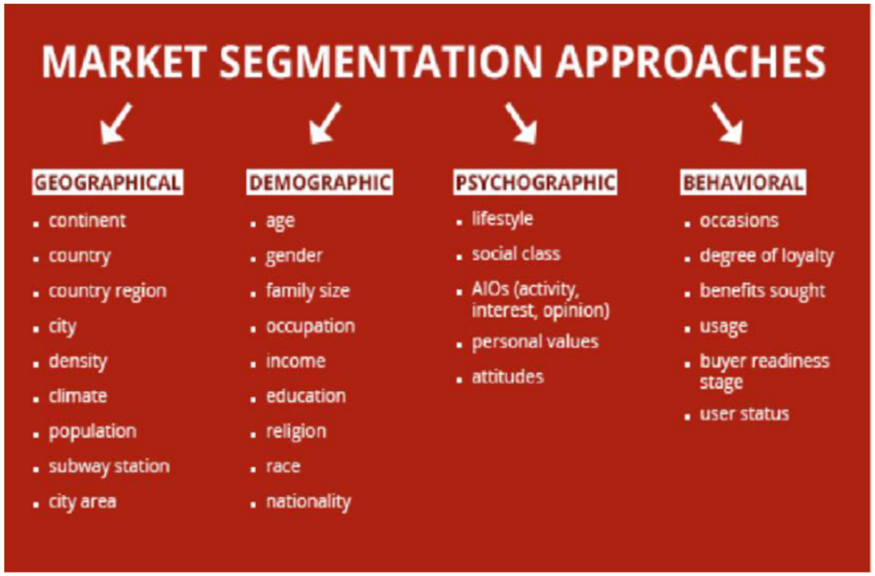
Clustering

- Given $N \times d$ data matrix: $\boldsymbol{X}$
  - Each row is one sample, x_n
- Problem: Group data into K clusters
- Mathematically:
  - Assign each sample to a cluster
  - Assign $\sigma_n \in \{1, \ldots, K\}$: Cluster label for each sample
- Want samples in same cluster to be "close"
  - $\|x_n - x_m\|$ is small when $\sigma_n = \sigma_m$



Clustering

- Clustering has many applications
  - Any time you want to segment data
  - Uncovering latent discrete variables
- Examples:
  - Segmenting sections of an image
  - Segmenting customers in market data



From: Market segmentation possibilities in the tourism market context of South Africa

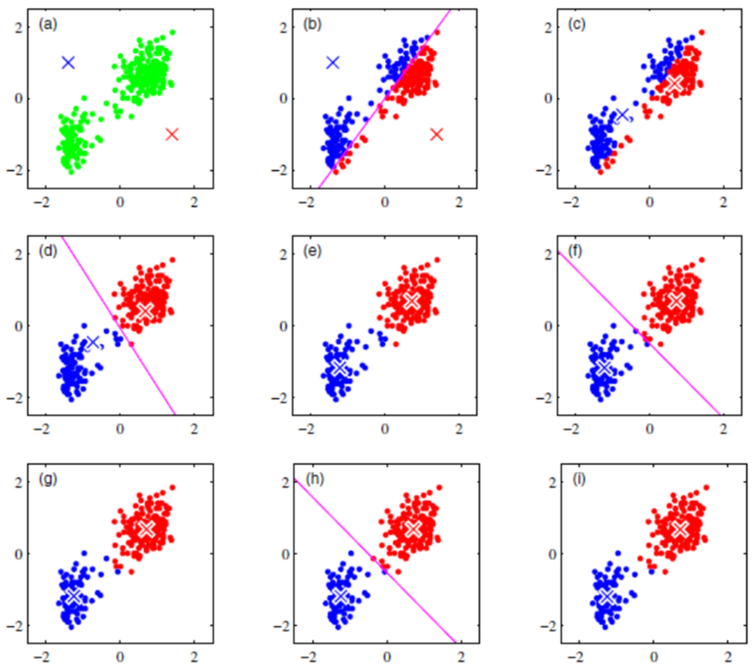
K-means

- A simple iterative algorithm to determine:
  - μ_i = mean of each cluster (hence, the name K-means)
  - $\sigma_n \in \{1, \ldots K\}$ = cluster that data point x_n belongs to
- Step 0: Start with guess at σ_n
- Step 1: Update mean of each cluster: $\mu_i =$ average of x_n s.t. $\sigma_n = i$
- Step 2: Update cluster membership:

$$\sigma_n = \arg\min_i \|x_n - \mu_i\|^2$$

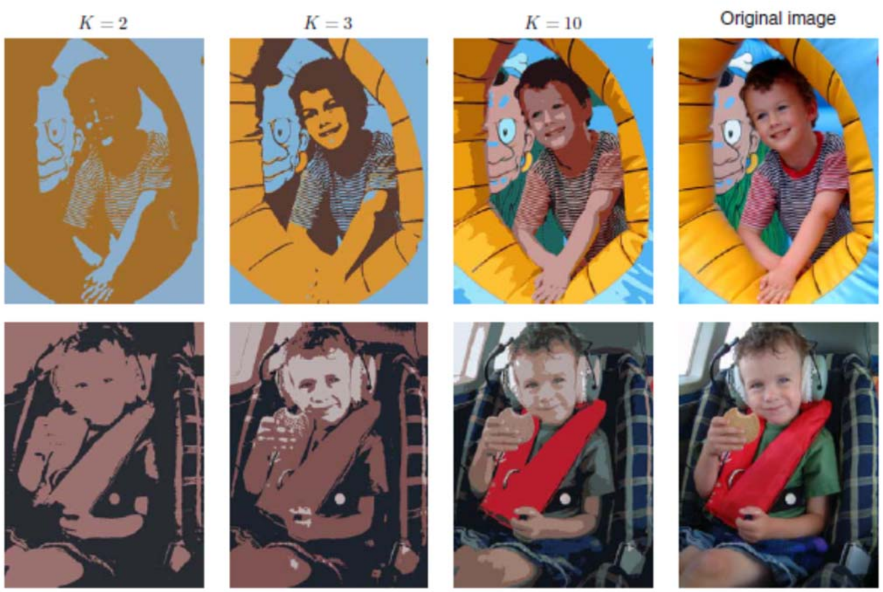
  - Selects cluster with closest mean
- Return to step 1

K-Means illustrated



- From Bishop, Chapter 9.
- K-Means on “old faithful” data set

Image Segmentation



- Also from Bishop.
- Use K-means on the RGB values (dimension = 3)

Convergence

- Will always converge to a "local" minima of cost function

$$J = \sum_{i=1}^{K} \sum_{n=1}^{N} r_{ni} \|x_n - \mu_i\|^2$$

  - Subject to $r_{ni} = 0$ or 1 and $\sum_i r_{ni} = 1$
- K-means alternately decreases J
  - Proof on board
- But, can get stuck in a local minima
  - May need good selection of initial condition

Distance measures

- Distance measures
  - How do measure similarity between samples?
  - Above algorithms used squared distance $||x_n - x_m||$

- Many possibilities
  - How to represent data as a vector?
  - Should you normalize entries?
  - What distance metric should you use?

Initialization

- Initialization:
  - Final limit of K-means depends on initial condition
  - May obtain poor clustering with bad initial condition
- Possible solutions:
  - K-means++: http://ilpubs.stanford.edu:8090/778/1/2006-13.pdf
  - Provides good initial condition based on data
  - Multiple initial starts

Outline

- Motivating Example: Document clustering
- K-means
- K-means for document clustering
- Latent semantic analysis
- Gaussian Mixture models (GMMs)
- Expectation Maximization (EM) fitting of GMMs
- Convergence of EM

Bag of Words

Document 1

The quick brown fox jumped over the lazy dog's back.

Document 2

Now is the time for all good men to come to the aid of their party.

Term	Document 1	Document 2
aid	0	1
all	0	1
back	1	0
brown	1	0
come	0	1
dog	1	0
fox	1	0
good	0	1
jump	1	0
lazy	1	0
men	0	1
now	0	1
over	1	0
party	0	1
quick	1	0
their	0	1
time	0	1

Stopword List
for
is
of
the
to

- Document is natively text
- Must represent as a numeric vector
- Represent by word counts
  - Enumerate all words
  - Each document is count of frequencies
- Stopwords

Discussion Questions

- Is the absolute number of times a word appears the correct metric?

- What about the length of the document?
- What about the frequency of the word?
- What words “matter”?

Term Frequency – Inverse Document Frequency

- Use TF-IDF weight for vectors:

$$X[n,i] = \quad TF_{i,n} \quad \times \quad IDF_i$$

Document weight vector

Term frequency

$$= \frac{\text{num times word } i \text{ in doc } n}{\text{total num words in doc } n}$$

Inverse doc frequency

$$= \log\left[\frac{\text{Total num docs in corpus}}{\text{Num docs with word } i}\right]$$

Computing TF-IDF in Python

- Can compute the TF-IDF using sklearn functions

```
print("Extracting features from the training dataset using a sparse vectorizer")
t0 = time()
vectorizer = TfidfVectorizer(max_df=0.5, max_features=opts.n_features,
                             min_df=2, stop_words='english',
                             use_idf=opts.use_idf)
X = vectorizer.fit_transform(dataset.data)
print("done in %fs" % (time() - t0))
print("n_samples: %d, n_features: %d" % X.shape)
print()
```

```
Extracting features from the training dataset using a sparse vectorizer
done in 1.451549s
n_samples: 3387, n_features: 10000
```

Typical TF-IDF scores

weimar	0.565396
pov	0.518174
renderers	0.183033
univesa	0.178595
und	0.174842
fuer	0.171591
true	0.159214
raytracing	0.150534
displaying	0.140240
ve	0.139752
options	0.134309
rendering	0.133027
driver	0.129544
happens	0.122540
colors	0.119138
card	0.113776
display	0.108457
germany	0.108231
tried	0.106282
color	0.103717
anybody	0.100397
correct	0.100234
isn	0.084694
got	0.081865
keywords	0.081865
try	0.080601
help	0.078058
nntp	0.044277
host	0.043985
posting	0.042608

- Code to display terms with highest scores

```
xi = X[doc_ind,:].todense()
term_ind = xi.argsort()[:, ::-1]
xi_sort = xi[0,term_ind]
terms = vectorizer.get_feature_names()

for i in range(30):
    term = terms[term_ind[0,i]]
    tfidf = xi[0,term_ind[0,i]]
    print('{0:20s} {1:f} '.format(term, tfidf))
```

Running K-Means

- Use Python built-in function

```
km = KMeans(n_clusters=true_k, init='k-means++', max_iter=100, n_init=1,
                verbose=opts.verbose)
```

```
print("Clustering sparse data with %s" % km)
t0 = time()
km.fit(X)
print("done in %0.3fs" % (time() - t0))
print()
```

```
Clustering sparse data with KMeans(algorithm='auto', copy_x=True, init='k-means++', max_iter=100,
    n_clusters=4, n_init=1, n_jobs=1, precompute_distances='auto',
    random_state=None, tol=0.0001, verbose=True)
Initialization complete
Iteration  0, inertia 6464.681
Iteration  1, inertia 3297.729
Iteration  2, inertia 3281.166
Iteration  3, inertia 3277.920
Iteration  4, inertia 3276.435
Iteration  5, inertia 3274.901
Iteration  6, inertia 3273.224
Iteration  7, inertia 3271.565
Iteration  8, inertia 3270.516
```

Plotting the Results

- Most important words in each cluster
  - Highest weights in cluster centers

```
order_centroids = km.cluster_centers_.argsort()[:, ::-1]
for i in range(true_k):
    print("Cluster %d:" % i, end='')
    for ind in order_centroids[i, :10]:
        print(' %s' % terms[ind], end='')
    print()
```

```
Cluster 0: graphics com university image posting thanks host nntp computer ac
Cluster 1: god com people don say jesus article think bible christian
Cluster 2: space nasa henry access digex toronto pat alaska gov shuttle
Cluster 3: sandvik sgi livesey com kent apple keith newton solntze wpd
```

Confusion Matrix

- Estimated clusters vs. true categories
- Can you see where it got confused?

```
labelkm = km.labels_
from sklearn.metrics import confusion_matrix
C = confusion_matrix(labels,labelkm)

Csum = np.sum(C,axis=0)
Cnorm = C / Csum[None,:]
print(Cnorm)
```

```
[[ 0.02664797  0.5559633   0.          0.6512605 ]
 [ 0.67461431  0.00458716  0.00947867  0.        ]
 [ 0.24263675  0.01651376  0.98420221  0.        ]
 [ 0.05610098  0.42293578  0.00631912  0.3487395 ]]
```

```
dataset.target_names
```

```
['alt.atheism', 'comp.graphics', 'sci.space', 'talk.religion.misc']
```

An Example “Wrong” cluster

```
Actual newsgroup: talk.religion.misc
Most common newsgroup in cluster:  alt.atheism

From: skinner@sp94.csrd.uiuc.edu (Gregg Skinner)
Subject: Re: Davidians and compassion
Reply-To: g-skinner@uiuc.edu
Organization: UIUC Center for Supercomputing Research and Development
Lines: 26

sandvik@newton.apple.com (Kent Sandvik) writes:

>In article <1993Apr20.143400.569@ra.royalroads.ca>, mlee@post.RoyalRoads.ca
>(Malcolm Lee) wrote:
>> Do you judge all Christians by the acts of those who would call
>> themselves Christian and yet are not?  The BD's contradicted scripture
>> in their actions.  They were NOT Christian.  Simple as that.  Perhaps
>> you have read too much into what the media has portrayed.  Ask any
>> true-believing Christian and you will find that they will deny any
>> association with the BD's.  Even the 7th Day Adventists have denied any
>> further ties with this cult, which was what they were.

>Well, if they were Satanists, or followers of an obscure religion,
>then I would be sure that Christians would in unison condemn and
>make this to a show case.

You might be sure, but you would also be wrong.

>And does not this show the dangers with religion -- in order
>word a mind virus that will make mothers capable of letting
>their small children burn to ashes while they scream?

I suspect the answer to this question is the same as the answer to,
"Do not the actions of the likes of Stalin show the dangers of
atheism?"
```

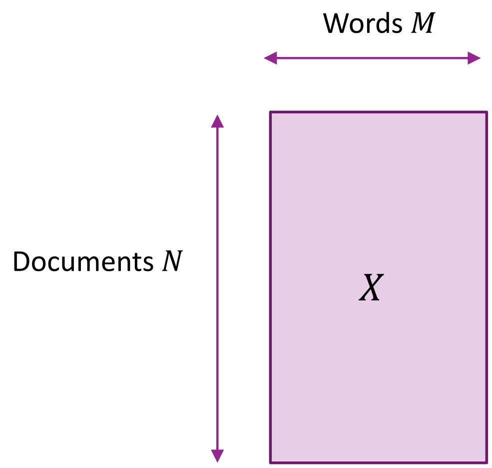
- ❑Post is from talk.religion.misc
- ❑Placed in cluster with mostly alt.atheism

Outline

- Motivating Example: Document clustering
- K-means
- K-means for document clustering
- Latent semantic analysis
- Gaussian Mixture models (GMMs)
- Expectation Maximization (EM) fitting of GMMs
- Convergence of EM

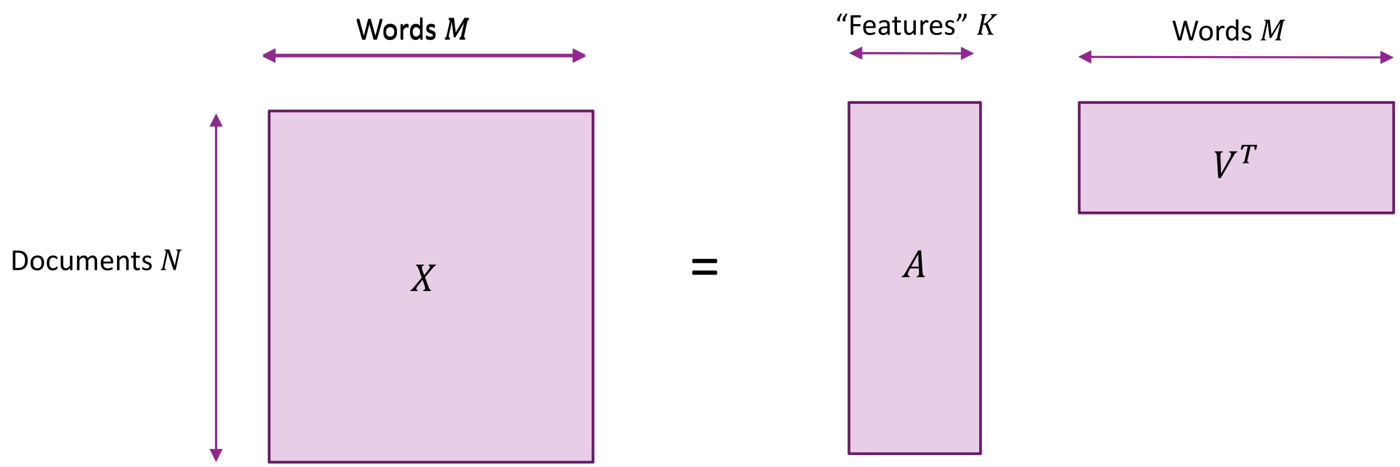
Need for Dimensionality Reduction

- Term-document matrix X is large
  - $N =$ document $\times$ M words in vocabulary
- M is large
  - Can be 10^6 in commercial systems
- Document represented by long sparse vector
- Inefficient
- Need dimensionality reduction



Latent Semantic Analysis

- LSA = PCA on term-document matrix
- $X \approx USV^T = AV^T, \quad A = US$
  - U, S, V^T computed by low-rank SVD



LSA Interpretation

- Each PC represents a "topic" or "concept"
- PC decomposition:

$$X[n,i] \approx \sum_{k=1}^{K} A[n,k]V[i,k]$$

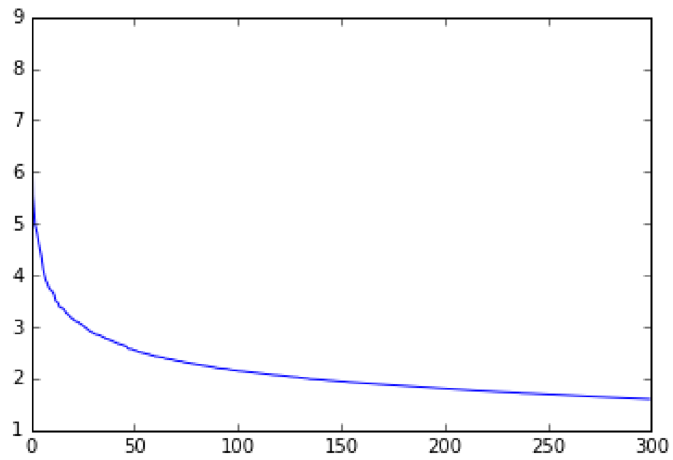
  - $A[n,k]$ = component of topic k in document n
  - $V[i,k]$ = component of word i in topic k

- Learn much more (in advanced ML class):
  - Word and document embeddings
  - Latent Dirchelet Allocation
  - ...

Perform LSA on NewsGroup

```
import scipy.sparse.linalg
U1,S1,V1 = scipy.sparse.linalg.svds(X,k=300)
```

```
plt.plot(S1[::-1])
```



- Use sparse SVD
- Much faster
- Concentration of variance in small number of PCs
- More interesting results in larger corpi

Outline

- Motivating Example: Document clustering
- K-means
- K-means for document clustering
- Latent semantic analysis
- Gaussian Mixture models (GMMs)
- Expectation Maximization (EM) fitting of GMMs
- Convergence of EM

Mixture Models

- Sometimes useful to have a probabilistic model of clustering
- Random variable $z \in \{1, \ldots, K\}$
  - Some discrete event with PMF: $P(z = i)$
  - Typically not observed directly
  - Called a latent variable
- Observed variable x, can be continuous
  - Probability depends on z, $p(x|z = i)$
  - One PDF per state $z = i$
  - Each PDF is called a component

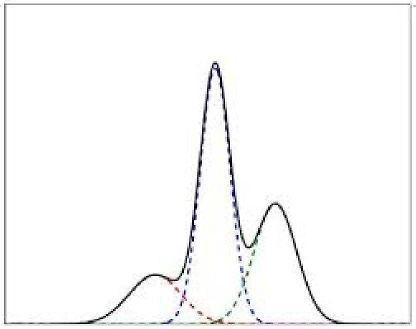
Examples

- Many data occurs from underlying discrete states
- Example 1: Size of a webpage
  - z = content of the webpage, e.g. number of images
- Example 2: Speech
  - z = phoneme the speaker is saying
- Example 3: Image
  - x = RGB values of a pixel or region of pixels
  - $z =$ one a small number of objects the pixel is part of

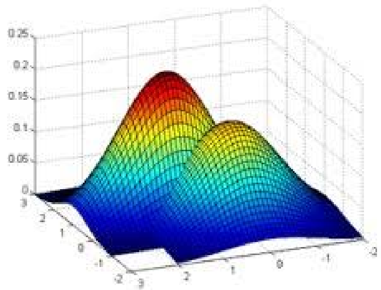
Gaussian Mixture Models

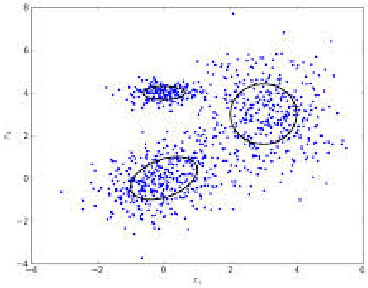
- Each $p(x|z=i)$ is a Gaussian
- Parametrized by:
  - $q_i = P(z=i)$ = Probability of each component
  - $\mu_i = E(x|z=i), P_i = var(x|z=i)$
    mean and variance in each component
- Can be vector valued
- Distribution of x can be computed via total probability
  - PDF $p(x) = \sum p(x|z=i)P(z=i)$
  - CDF $F(x_0) = \sum P(x \leq x_0|z=i)P(z=i)$

Visualizing GMMs



- 1d model with $K = 3$ components





- PDF for 2d GMM with $K = 2$ components
- Random points from a GMM with $K = 3$ components

Expectation and Variance

- Can compute expectation and variance by total probability
  - Expectation: $\mu = E(x) = \sum q_i \mu_i$
  - Variance:

$$var(x) = \sum_i P_i \qquad + q_i(\mu_i - \mu)(\mu_i - \mu)^T$$

Variance within component

Variance between components

- Proof on board

Determining the Component

- Given x, can we determine z
- Use Bayes' rule:

$$P(z = i|x) = \frac{P(x|z = i)q_i}{\sum_k P(x|z = k)q_k}$$

- Example: Scalar Gaussian with two components: (μ_1, σ^2) and (μ_2, σ^2), $q_1 = q_2 = 0.5$

$$P(z = 1|x) = \frac{e^{-(x-\mu_1)^2/2\sigma^2}}{e^{-(x-\mu_1)^2/2\sigma^2} + e^{-(x-\mu_2)^2/2\sigma^2}} = \frac{1}{1 + e^{-a(x-b)}}$$

  - Sigmoid shape
  - $a = (\mu_2 - \mu_1)/\sigma^2 =$ SNR
  - $b = (\mu_2 + \mu_1)/2 =$ center

Outline

- Motivating Example: Document clustering
- K-means
- K-means for document clustering
- Latent semantic analysis
- Gaussian Mixture models (GMMs)
- Expectation Maximization (EM) fitting of GMMs
- Convergence of EM

Maximum Likelihood Estimation

- Unknown parameters in GMM: $\theta = (q_1, \ldots, q_K, \mu_1, .., \mu_K, P_1, \ldots, P_K)$
- Data $x = (x_1, \ldots, x_N)$
- Negative log likelihood:

$$L(\theta) = -\ln p(x|\theta) = -\sum_{n=1}^{N} \ln \left[\sum_{i=1}^{K} q_i N(x_n|\mu_i, P_i) \right]$$

- ML estimation:

$$\hat{\theta} = \arg\min L(\theta)$$

  - No simple way to directly optimize
  - Likelihood is non-convex

Expectation Maximization

- Consider joint probability

$$p(x_n, z_n = i|\theta) = q_i N(x_n|\mu_i, P_i)$$

- Iterative procedure:
  - Generates a sequence of estimates $\hat{\theta}^0, \hat{\theta}^1, \ldots$

- Attempts to approach MLE

$$\hat{\theta}^k \to \arg\min_{\theta} L(\theta)$$

EM Steps

- E-step: Estimate the latent variables
  - Find the posterior of the latent variables given $\hat{\theta}^k$: $P(z|x,\theta=\hat{\theta}^k)$
  - Compute auxiliary function

$$Q(\theta,\hat{\theta}^k) := E\left[\ln p(x,z|\theta)|\hat{\theta}^k\right] = \sum_z \ln p(x,z|\theta)\, P(z|x,\theta=\hat{\theta}^k)$$

- M-step: Update parameters

$$\hat{\theta}^{k+1} = \arg\max_{\theta} Q(\theta,\hat{\theta}^k)$$

- Will discuss convergence later

E-Step for a GMM

Finding the posterior

- Given parameters q_i, μ_i, P_i
- Find posterior by Bayes rule

$$\gamma_{ni} = P(z_n = i|x) = \frac{P(x_n|z_n = i)q_i}{\sum_k P(x_n|z_n = k)q_k} = \frac{N(x_n|\mu_i, P_i)q_i}{\sum_k P(x_n|\mu_k, P_k)q_k}$$

- A “soft” selection

E-Step for a GMM

- Auxilliary function separates

$$Q\left(\theta, \hat{\theta}^k\right) = E\left[\ln p(x,z)|\hat{\theta}^k\right]$$

$$= \sum_{i=1}^{K} \sum_{n=1}^{N} \gamma_{ni} \ln P(x_n, z_n = i)$$

$$= \sum_{i=1}^{K} \sum_{n=1}^{N} \gamma_{ni} [\ln q_i + \ln N(x_n|\mu_i, P_i)]$$

M-Step for the GMM

- Maximize $Q(\theta, \hat{\theta}^k)$
- Update for q_i (proof on board)

$$q_i = \frac{N_i}{\sum_j N_j}, \qquad N_i = \sum_n \gamma_{ni}$$

- Update for μ_i

$$\mu_i = \frac{1}{N_i} \sum_n \gamma_{ni} \, x_n$$

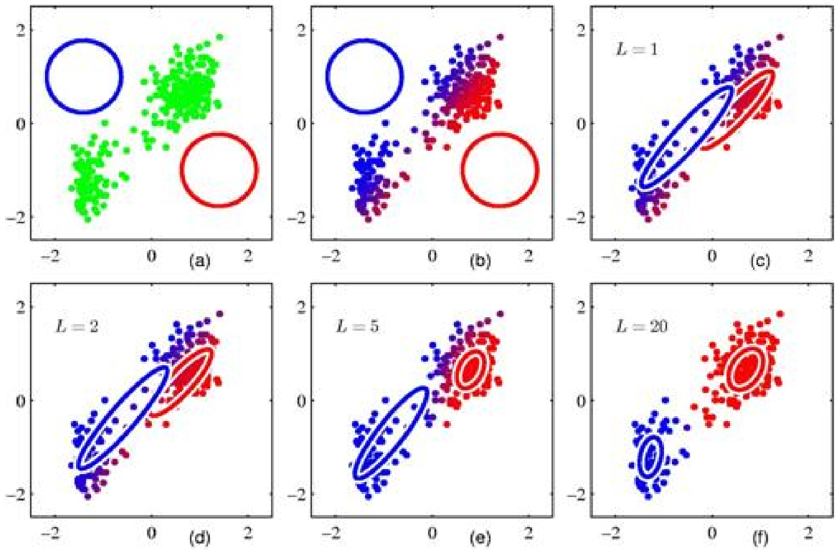
- Update for P_i

$$P_i = \frac{1}{N_i} \sum_n \gamma_{ni} \, (x_n - \mu_i)(x_n - \mu_i)\hat{}T$$

Relation to K-Means

- EM can be seen as a “soft” version
  - In K-Means: $\gamma_{ni} = 1$ or 0
- Variance
  - In K-means: $P_i = I$
  - In EM, this is estimated
- EM provides “scaling” of various dimensions

EM Illustrated



- Simple example with K=2 clusters
- Dimension = 2
- Can have bad convergence from poor initial condition

Outline

- Motivating Example: Document clustering
- K-means
- K-means for document clustering
- Latent semantic analysis
- Gaussian Mixture models (GMMs)
- Expectation Maximization (EM) fitting of GMMs
- Convergence of EM

Majorization Minimization

- Suppose we wish to minimize $f(\theta)$
- MM algorithm: find a majorizing function $F(\theta, \theta^k)$:
  - $f(\theta^k) = F(\theta^k, \theta^k)$
  - $f(\theta) \leq F(\theta, \theta^k)$ for all θ
- Take $\theta^{k+1} = \arg\min_{\theta} F(\theta, \theta^k)$ (minimize majorization)
- Theorem: $f(\theta^{k+1}) \leq f(\theta^k)$
- Proof:

$$f(\theta^{k+1}) \leq F(\theta^{k+1}, \theta^k) \leq F(\theta^{k+1}, \theta^k) \leq f(\theta^k)$$

Gradient Descent as a MM

- Find $\alpha \geq f''(\theta)$
- Define

$$F(\theta, \theta^k) = f(\theta^k) + \nabla f(\theta^k)(\theta - \theta^k) + \frac{\alpha}{2}\left\|\theta - \theta^k\right\|^2$$

- By Taylor's theorem, this is a majorizing function
- Gradient descent:

$$\theta^{k+1} = \arg\min_{\theta} F(\theta, \theta^k) = \theta^k - \frac{1}{\alpha}\nabla f(\theta^k)$$

Convergence of EM

- $p(z|x,\theta) = p(x,z|\theta)/p(x|\theta)$
- $J(\theta) = -\ln p(x|\theta) = -\ln p(x,z|\theta) + \ln p(z|x,\theta)$
- $J(\theta) = -E\left[\ln p(x,z|\theta)\,|\theta^k\right] + E\left[\ln p(z|x,\theta)\,|\theta^k\right] = -Q\left(\theta,\theta^k\right) - H(\theta,\theta^k)$
- By Gibbs Theorem: $H\left(\theta,\theta^k\right) = -E\left[\ln p(z|x,\theta)\,|\theta^k\right] \geq 0$ with $H\left(\theta^k,\theta^k\right) = 0$
  - $Q\left(\theta,\theta^k\right)$ is a majorizing function
- EM is the MM algorithm applied to $Q\left(\theta,\theta^k\right)$
- Therefore, $J\left(\theta^{k+1}\right) \geq J(\theta^k)$
- Algorithm may get stuck in local maxima