Machine Learning in Three Lectures

Sach Mukherjee

DZNE, Bonn

Lecture II: outline

- Focus on *unsupervised learning*
- Wide class of problems where there are either no labels available in the data or where the goal is less explicit
- Specifically focus on *clustering* (and connections to related models), *dimensionality reduction* and *matrix completion*
- General way to think about such models is as having hidden or latent variables or some form of "simple" structure

Clustering

Recap: classification

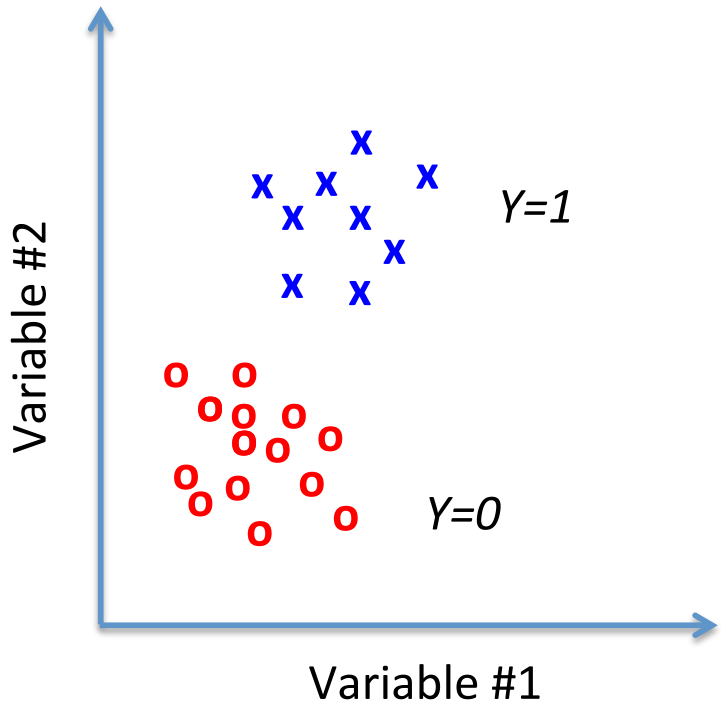
- Task: given data

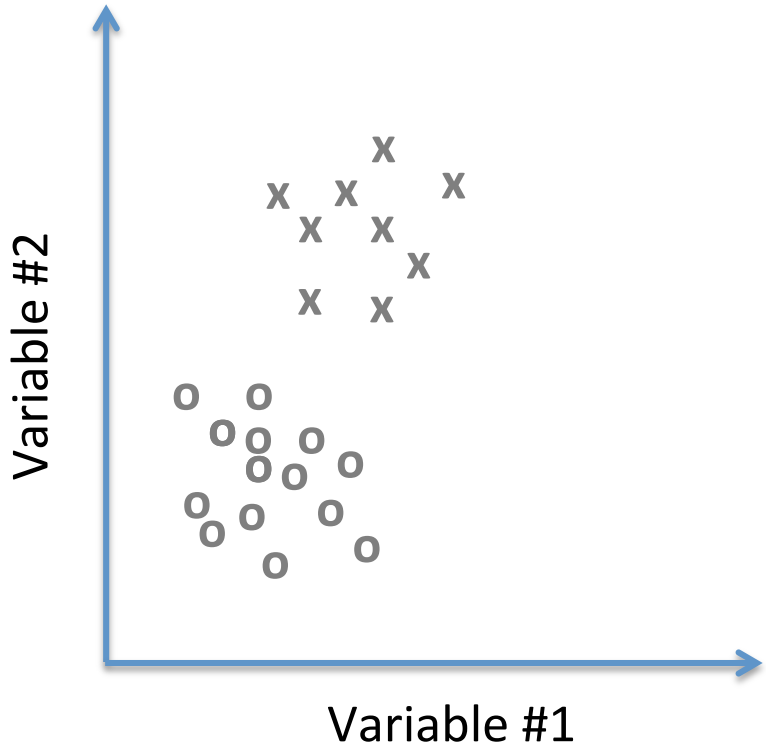
$$(x_i, y_i)_{i=1\ldots n},\ x_i \in \mathbb{R}^p,\ y \in \{0, 1\},$$

  wanted to learn a function $\hat{f} : \mathbb{R}^p \to \{0, 1\}$

- Wanted $\hat{f}$ accurate in the sense that for a *new* pair (X', Y'), $Pr(\hat{f}(X') = Y')$ is high
- Saw two specific approaches via conditional probability distributions and via a logistic function

- What if labels are not available *at all* ? That is, initial data is only $X = [x_1 \ldots x_n]^{\mathrm{T}}$ (no y's)?


Variable #2
Variable #1
Y=1
Y=0


Variable #2
Variable #1

Clustering

- This is the classical "clustering" problem, direct latent variable analogue of classification
- Task: given data

$$(x_i)_{i=1\ldots n},\ x_i \in \mathbb{R}^p$$

  assign each point to one of K classes ("clusters")
- First we will look at a classical method called *K-means clustering*, then at a probabilistic formulation with a hidden variable interpretation, and then at the connection between the two

Applications of clustering

- Vast number of applications of clustering due to the fact that we can often make high-dimensional measurements but have only rough ideas, if any, of how to categorize objects
- Medicine/biology: discovering new subtypes of disease, often clusters correspond to different underlying biology that was not previously known, or (usefully) contradict disease classes that were more historical than truly evidence-based
- Retail: clustering to identify specific customer subtypes to then target marketing/product development etc.
- Signal processing: to compress data, by so-called vector quantization

$K = 3$

$K = 10$

Original image

(Bishop, 2006)

K-means clustering

- Task: given data
$$(x_i)_{i=1\ldots n},\ x_i \in \mathbb{R}^p$$
and specified number of classes K assign each point to one of K classes ("clusters")

- As the name suggests, K-means parameterizes each cluster with a cluster specific mean $\mu_k \in \mathbb{R}^p$ and gives an assignment $C(i) \in \{1 \ldots K\}$ for each of the n points.

K-means clustering: objective function

- The idea is to choose the μ_k's and $C(i)$'s to minimize the distances between the sample points $(x_i)_{i=1\ldots n}$ and their assigned means.
- The objective function is

$$J(\{\mu_k\}, \{C(i)\}) = \sum_{k=1}^{K} \sum_{i:C(i)=k} \|x_i - \mu_k\|_2^2$$

- Notice that if μ_k's are known, the $C(i)$ that minimizes J is just to assign each point to its closest mean
- Similarly, if $C(i)$ is known, the optimal μ_k's are simply the means of only those points with $C(i) = k$

K-means clustering: optimization

- This suggests a simple alternating scheme. Initialize the μ_k's.

  (1) Keeping μ_k's fixed, for $i = 1 \dots n$ set

  $$C(i) \leftarrow \underset{k}{\operatorname{argmin}} \|x_i - \mu_k\|_2^2$$

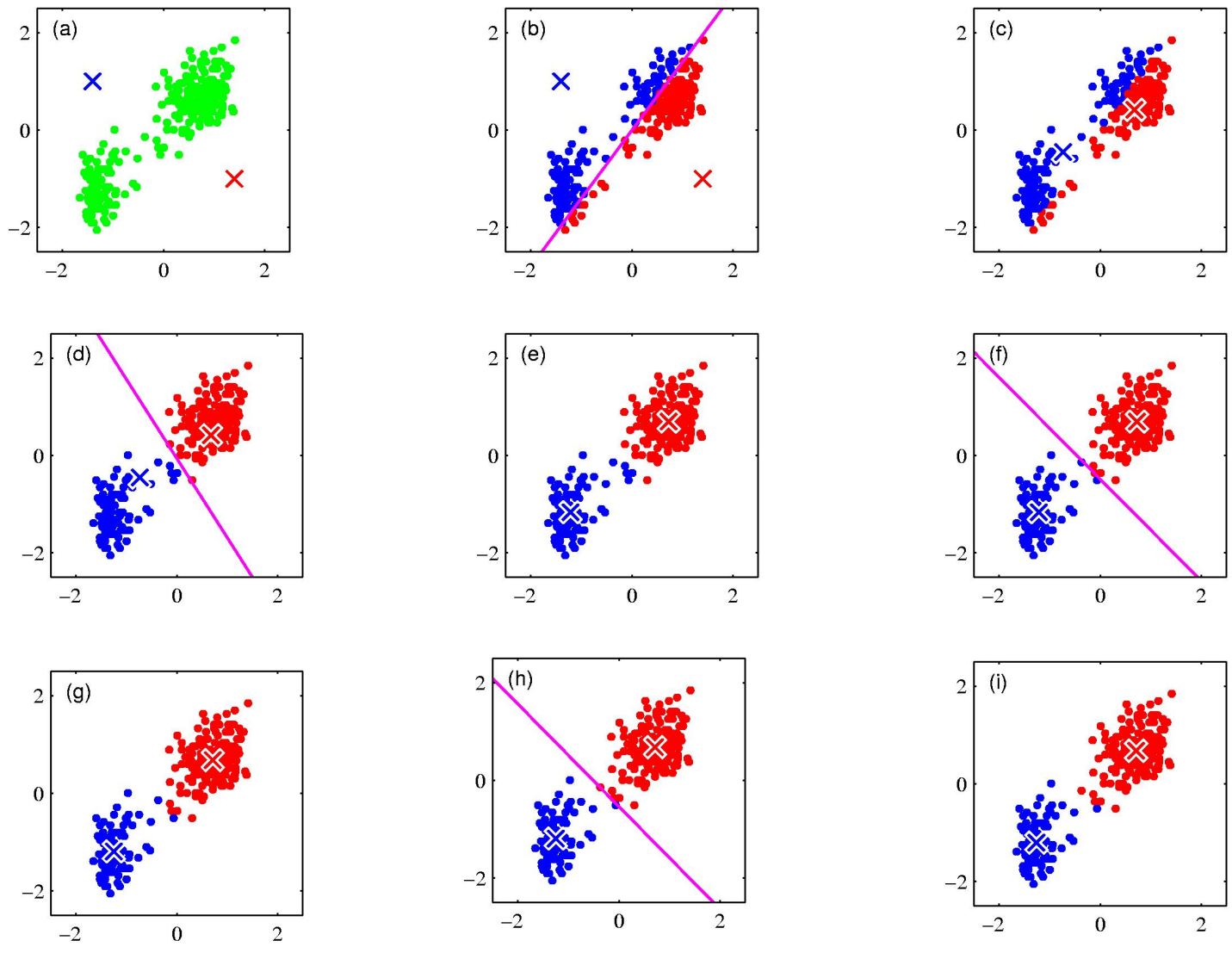
  (2) Then, keeping assignments $C(i)$ fixed for each k update the means

  $$\hat{\mu}_k \leftarrow \frac{1}{|\{i : C(i) = k\}|} \sum_{i:C(i)=k} x_i$$

- Repeat (1), (2) until convergence criterion is met

- This is the classical K-means algorithm
- Provides a local optimum, still (surprisingly?) effective and fast

K-means: example



(Bishop, 2006)

Mixture models: probability models for hidden groups

- Can be illuminating (and useful) to look at the clustering problem from a probabilistic point of view
- A *mixture model* g is a combination of component densities g_k

$$g(x) = \sum_{k=1}^{K} \pi_k \, g_k(x)$$

with $0 \leq \pi_k \leq 1$ and $\sum_k \pi_k = 1$

- Identifying the components k with clusters/classes, we can see that this is a general analogue to the classification model, but gives a "marginal" model for the X's only

Gaussian mixture models

- Consider mixture of K Gaussians

$$g(x) = \sum_k \pi_k N(x \mid \mu_k, \Sigma_k)$$

- Complete model parameter is $\theta = (\pi_k, \mu_k, \Sigma_k)_{k=1..K}$
- If we knew which point X_i came from which groups, estimation would simply a Gaussian per class/cluster
- But we don't, so as for K-means, we use an alternating scheme, using current estimates of θ to "soft assign" the points and then using these assignments to re-estimate the Gaussians

Gaussian mixture models: estimation

- First, initialize parameters $(\pi_k, \mu_k, \Sigma_k)_{k=1..K}$
- Next, compute probabilistic assignments (*responsibilities*) for each point i and each component k as

$$\gamma_{ik} \leftarrow \frac{\pi_k N(x_i \mid \mu_k, \Sigma_k)}{\sum_l \pi_l N(x_i \mid \mu_l, \Sigma_l)}$$

- Thinking of a latent $Z_i \in \{1 \dots K\}$ this is the current estimate of $Pr(Z_i = k \mid x_i)$, with π_k playing the role of the "prior" $Pr(Z_i = k)$

Gaussian mixture models: estimation

- Next, update Gaussian parameters using all data but weighted by these probabilities

$$\hat{\pi}_k \leftarrow \frac{1}{n} \sum_i \gamma_{ik}$$

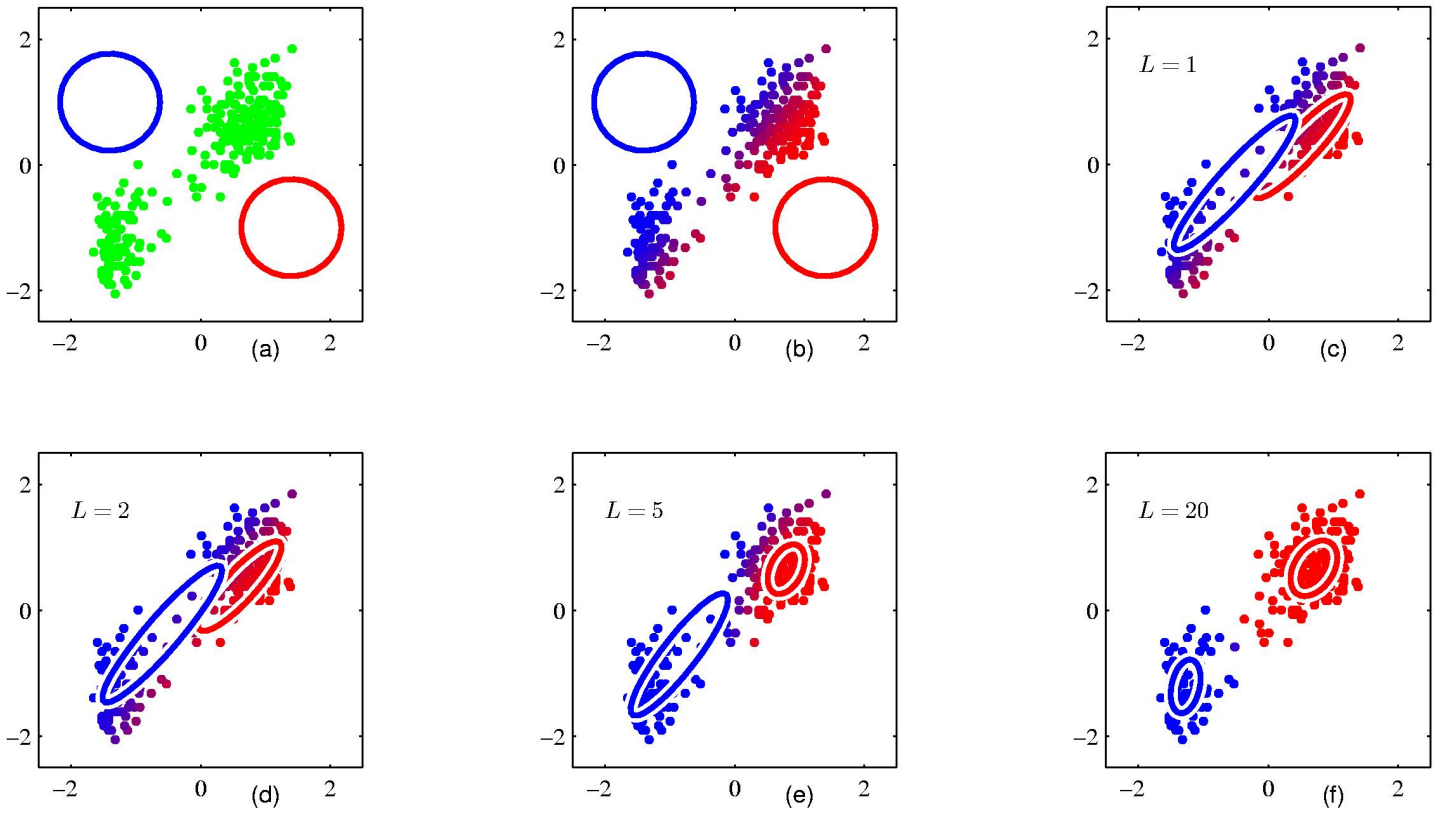
$$\hat{\mu}_k \leftarrow \frac{\sum_i \gamma_{ik} x_i}{\sum_i \gamma_{ik}}$$

$$\hat{\Sigma}_k \leftarrow \frac{\sum_i \gamma_{ik} (x_i - \hat{\mu}_k)(x_i - \hat{\mu}_k)^{\mathrm{T}}}{\sum_i \gamma_{ik}}$$

- Alternate until convergence or stopping criterion met

- This is an instance of a general algorithm known as the *Expectation-Maximization* or EM algorithm

EM: example



(Bishop, 2006)

K-means vs. mixture models

- K-means and mixture modelling noticeably similar
- K-means can be recovered as a limiting case of a Gaussian mixture model with all Gaussians isotropic, i.e. with all Σ_k's equal to $\sigma^2 I_p$ and taking $\sigma^2 \to 0$
- Then the probabilistic assignment becomes 0-1 and the EM steps become K-means

- Mixture models are much more flexible because they allow interesting covariance structure

- EM is highly general and notion of latent variables can be used in many settings, not just clustering

Dimensionality reduction

Dimensionality reduction

- Second broad class of unsupervised learning
- Task: given data $(x_i)_{i=1\ldots n}$, $x_i \in \mathbb{R}^p$, with p typically large, find a transformation $f : \mathbb{R}^p \to \mathbb{R}^q$, with $q \ll p$, such that the transformed data points $z_i = f(x_i)$ retain information/structure in the lower dimensional space
- f may be linear or nonlinear, many different potential criteria

Linear dimensionality reduction: PCA

- Consider linear projection down to one dimension, i.e. project data as $Z = Xw_1$, where w_1 is a unit-length p-vector. Want to retain as much variance as possible after projection
- Maximize projected variance, i.e. objective

$$J(w_1) = \underbrace{w_1^{\mathrm{T}} S w_1}_{=\mathrm{Var}(Z)} + \lambda_1(1 - w_1^{\mathrm{T}} w_1)$$

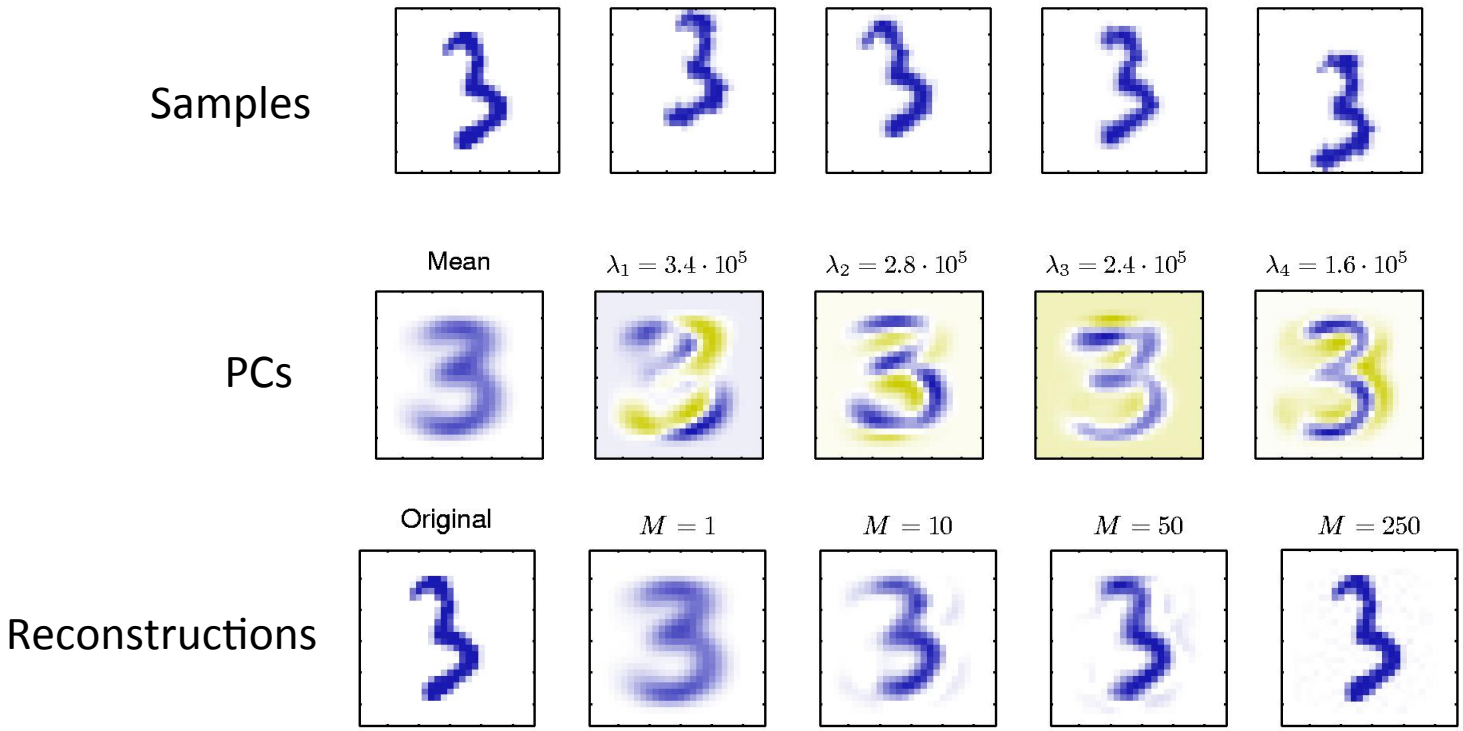
  with $S = \frac{1}{n} X^{\mathrm{T}} X$ being the sample covariance. The constraint enforces unit length of w_1
- Solution satisfies $Sw_1 = \lambda w_1$, i.e. w_1 is eigenvector of sample covariance
- Projected variance is $w_1^{\mathrm{T}} S w_1 = \lambda$, hence should choose eigenvector with largest eigenvalue

Linear dimensionality reduction: PCA

- For projection down to q dimensions, subsequent directions follow similar argument, variance maximizing projection is $Z = XW$, with $W = [w_1 \dots w_q]$, w_j's being q eigenvectors with largest eigenvalues
- This is the variance-maximizing linear projection into a q-dimensional subspace
- This approach is known as (classical) *principal components analysis* or PCA
- Regularized variants are possible, simply by augmenting the objective with additional constraints
- In practice – and despite some concerns from high-dimensional theory – PCA is often very effective

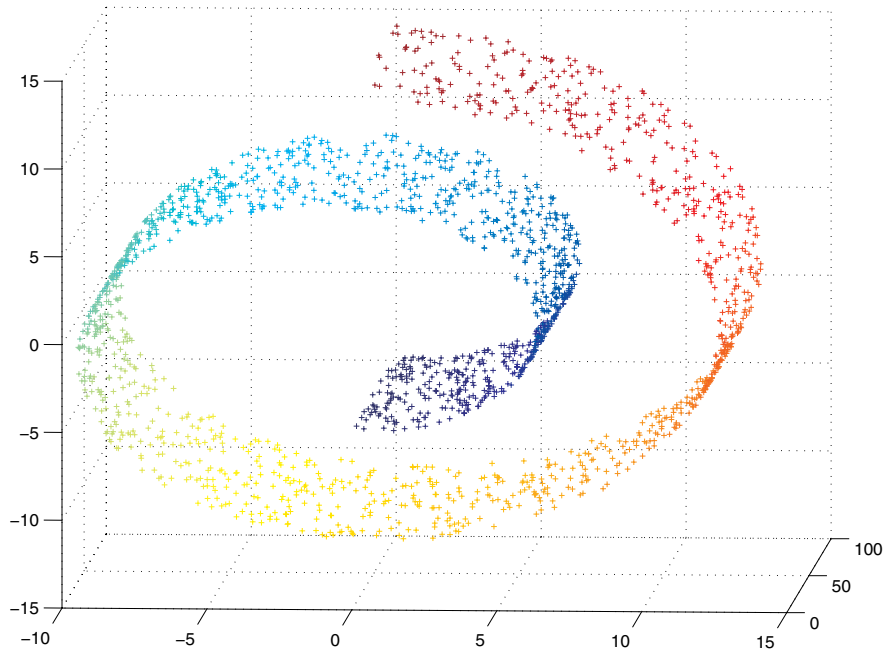
PCA: example

Here, the data are images, treated as vectors with $p = 10^4$



(Bishop, 2006)

Nonlinear dimensionality reduction



Nonlinear dimensionality reduction

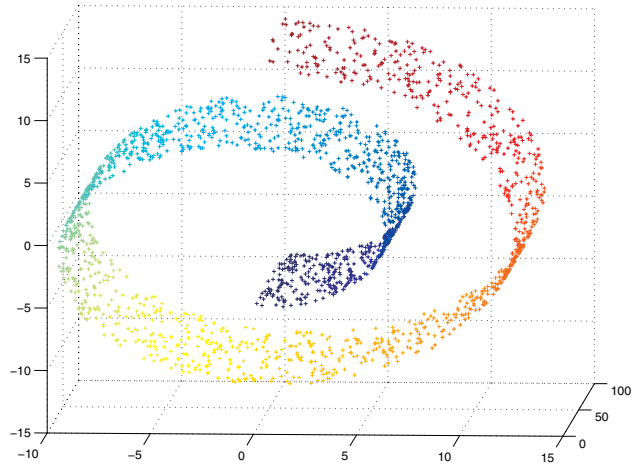
- Data may have nonlinear, low-dimensional structure that cannot be captured by a linear projection
- Wide range of nonlinear methods. One important class creates a weighted similarity graph between data points, i.e. a n-vertex graph with weights $W_{i,i'}$ capturing some notion of similarity
- Idea is to find a mapping f that preserves this structure. One way to do so is to minimize

$$\sum_{i,i'} \|f(x_i) - f(x_{i'})\|_2^2 \, W_{i,i'}$$

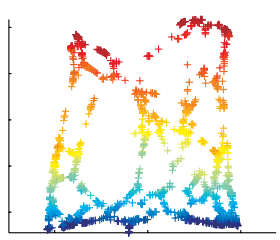
  s.t. identifiability constraints on f
- This can be solved as an eigenvalue problem wrt the so-called graph Laplacian derived from W. This is called a *Laplacian eigenmap*
- Details in construction of W can be influential

Nonlinear dimensionality reduction: example

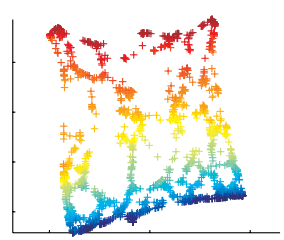
p=3 data



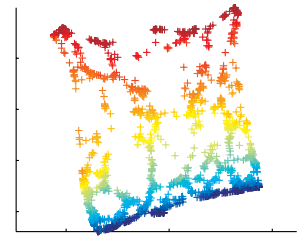
q=2 projection



N = 5 t = 5.0



N = 10 t = 5.0



N = 15 t = 5.0

(Belkin & Niyogi, Neural Computation, 2003)

Matrix completion

Matrix completion

- Another related problem is so-called *matrix completion* – idea is that you have an $n \times p$ matrix X that is incomplete but you assume that the underlying "complete" matrix Z has some specific structure
- Canonical example is Netflix customers-by-films matrix – here each film only seen by some of the customers, want to know which other customers *might* like to see the film, but large fraction is missing
- *Recommender systems* is a general term and *collaborative filtering* for the approach of pooling preferences to filter results
- Having the "completed" matrix would allow e.g. specific films to be targeted to specific customers

Matrix completion

- The available data are X. Some of the entries are unavailable
- Denote the completed matrix Z. This is a *latent matrix*, having the property that it should agree with X and satisfy some additional constraints
- Intuition is that in real matrices from such applications, there are correlations between rows and columns
- Suggests constraining Z to be *low rank*: if true, entries could be recovered by pooling the information across all rows and columns

Matrix completion

- One version of matrix completion can then be viewed as an optimization with objective

$$J(Z) = \|M \circ X - M \circ Z\|_F^2 + \lambda \sum_j \sigma(Z)_j$$

  where $\sigma(Z)_j$ are the singular values of Z and M is a $n \times p$ binary matrix with $M_{ij} = 1$ if X_{ij} is a non-missing entry

- Compare with sparse regression: the penalty induces sparsity – here low rank of Z – analogous to sparsity via ℓ_1-penalization of the regression model
- The problem can be efficiently solved and is widely used in practice

- However, note that (as with many ML methods) vanilla matrix completion may not be sufficient to fully address real problems and in practice many other approaches and heuristics are combined

Lecture II summary

- Today we looked at unsupervised learning problems
- Broad class that go beyond simply predicting something that is already in the available data towards all kinds of hidden structure in potentially high-dimensional data
- Exceptionally rich range of applications

Lecture II summary

- Recurrent themes of sparse/low-rank/low-dimensional structure, general idea is automatically finding representations that are "simple" in some sense
- Tools are mostly highly scalable, creative use can open up many practical problems