Unsupervised Learning

http://nlp.cs.berkeley.edu/comics.shtml

Slides from Hugo Larochelle, Geoffrey Hinton, and Yoshua Bengio

CSC411/2515: Machine Learning and Data Mining, Winter 2018

Michael Guerzhoy and Lisa Zhang

Supervised Learning

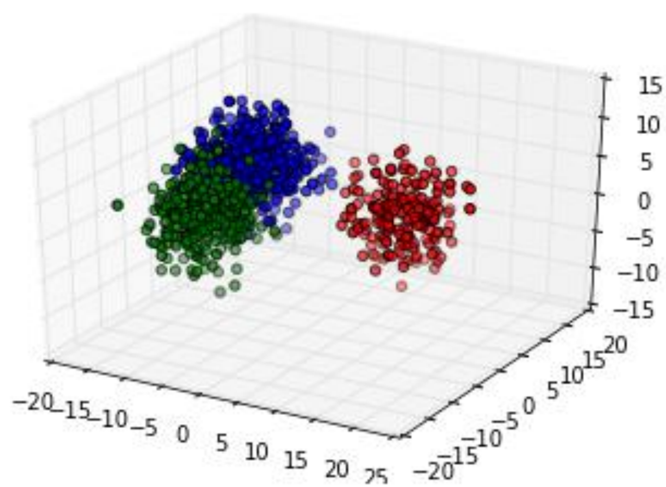
- "Programming by example"
  - Want to generalize patterns that we observe in the training set
    - Training example 1: $\mathrm{x}^{(1)} = \left(x_1^{(1)}, x_2^{(1)}, \dots., x_m^{(1)}\right)$ output: $y^{(1)}$
    - Training example 2: $\mathrm{x}^{(2)} = \left(x_1^{(2)}, x_2^{(2)}, \dots., x_m^{(2)}\right)$ output: $y^{(2)}$
    - Training example N: $\mathrm{x}^{(3)} = \left(x_1^{(N)}, x_2^{(N)}, \dots., x_m^{(N)}\right)$ output: $y^{(N)}$
- Can do pretty much anything if the training set is large enough

Unsupervised Learning

- No labels
- Goal: learn the structure in the data
  - Estimate the **probability** of a new unseen data point ("density estimation")
  - Extract useful **features** from the data by noticing patterns in the x's
  - **Generate** "fake" data that looks like the real data

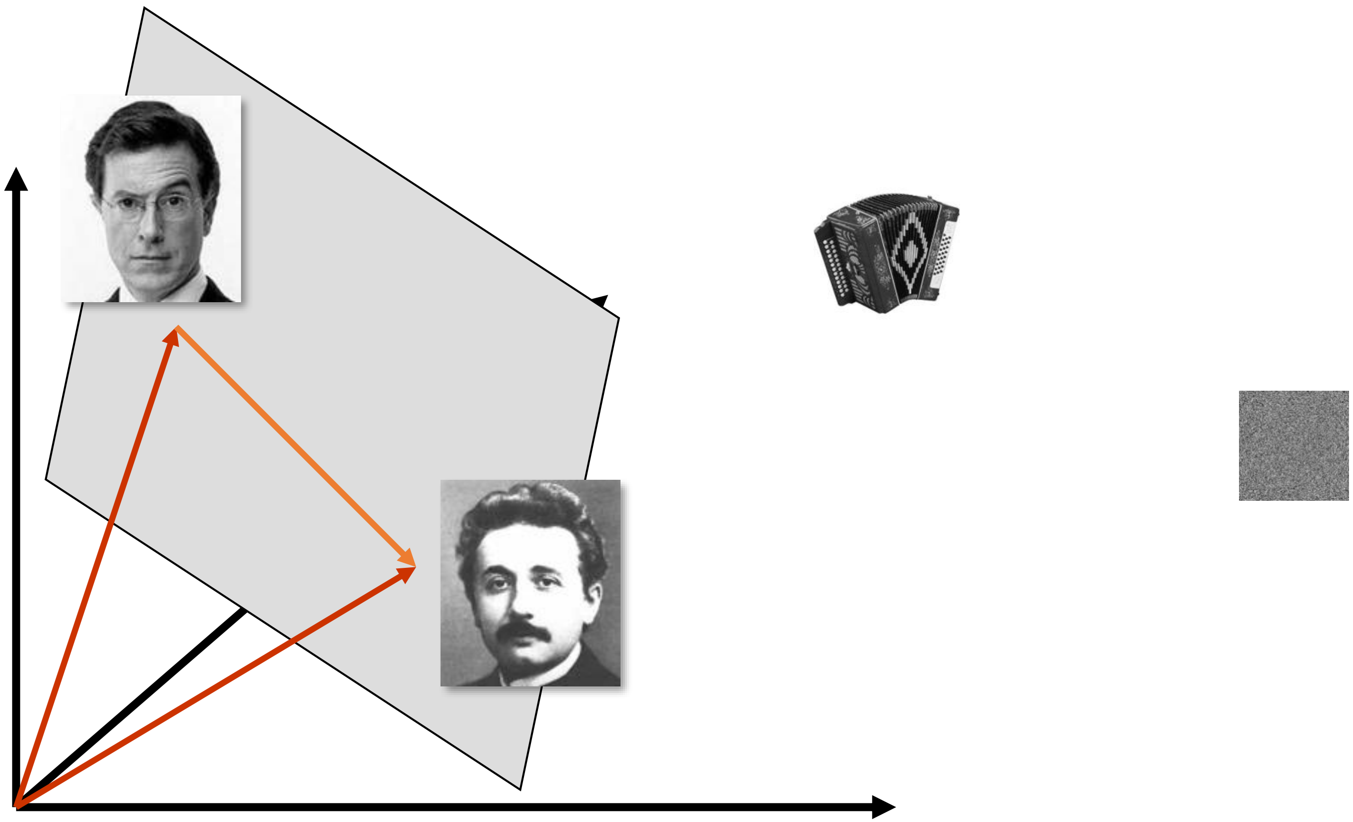
Mixture of Gaussians/k-Means

- Data: n-dimensional points in space
- Idea/assumption: the data are organized into clouds
  - Find the centres (μ_j) and sizes (Σ_j) of the clouds



PCA

- Data: n-dimensional points in space, centred around some point μ
- Idea: the centred x_i's (i.e., $(x_i - \mu)$'s) form a *subspace:* any x_i can be approximately reconstructed using $\widehat{x_i} \approx \mu + \alpha_1^i v_1 + .. + \alpha_k^i v_k$ for a small k
  - The points form a cloud that's not n-dimensional
  - Find a basis $v_1, \ldots, v_k$ (for a set k) s.t. $\sum_i \ (\hat{x}_i - x_i)^2$ is minimized
  - The $\alpha_1 \ldots \alpha_k$ encode most of the information about x
    - That's what lets us get a good reconstruction



- The set of faces is a "subspace" of the set of images
  - Suppose it is K dimensional
  - We can find the best subspace using PCA
  - This is like fitting a "hyper-plane" to the set of faces
    - spanned by vectors $\mathbf{v_1}, \mathbf{v_2}, \ldots, \mathbf{v_K}$
    - any face $\mathbf{x} \approx \overline{\mathbf{x}} + a_1\mathbf{v_1} + a_2\mathbf{v_2} + \ldots + a_k\mathbf{v_k}$

MoG+PCA

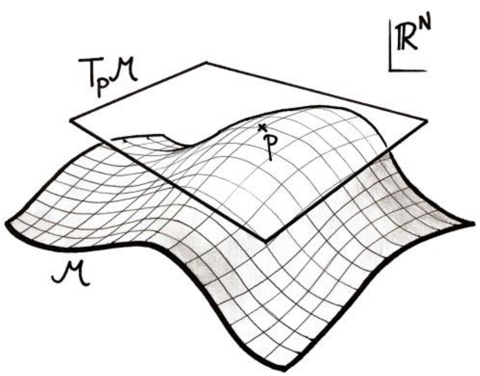
- First, find the clouds of points
- Then, apply PCA to each cloud separately

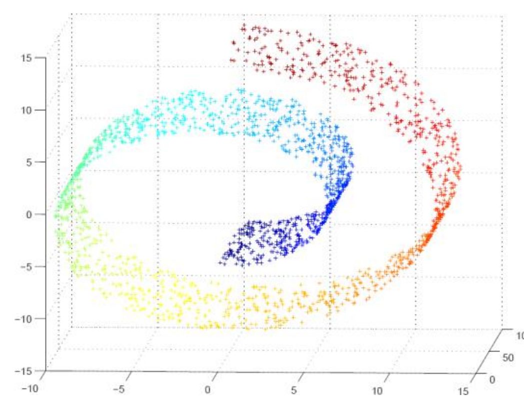
Other types of mixture models

- Mixture of NB Models:
  - Cluster emails into several groups that are characterized by word frequencies
- Mixture of Poisson:
  - Cluster (e.g.) queue lengths at Tim Hortons at different times into groups

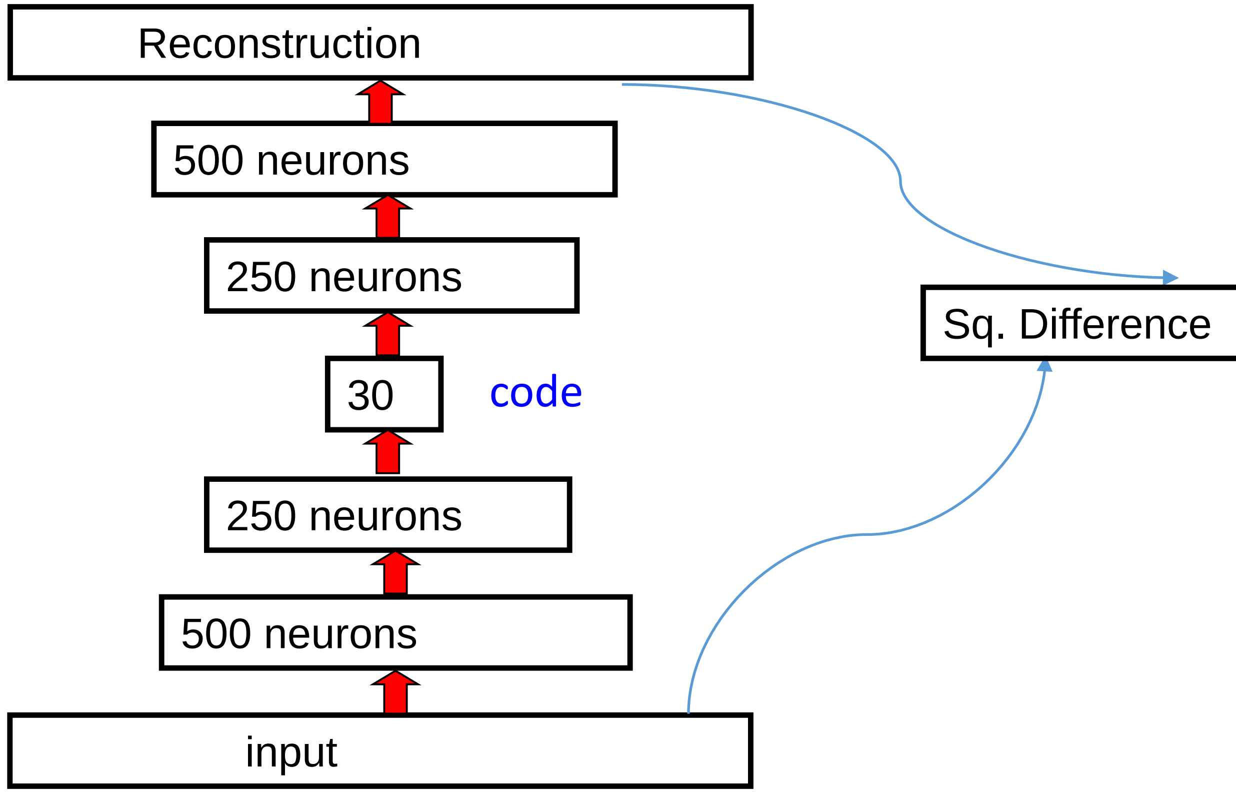
Autoencoders

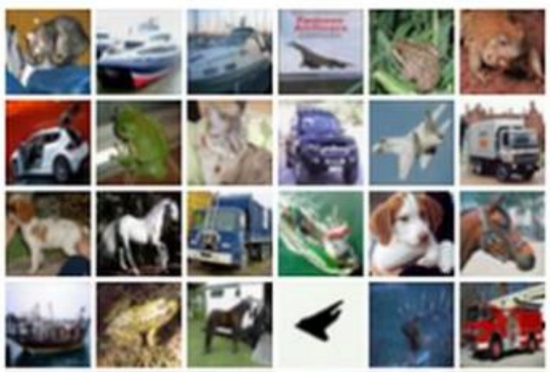
- Data: n-dimensional points in space
- Idea: the data x_i lies close to some manifold
  - Manifold: low-dimensional structures that "look like" hyperplanes locally
  - Can encode data in fewer dimensions
  - The coordinates of the low-dimensional representation represent the location on the manifold





Autoencoders





- Find the weights that produce as small a difference as possible between the input and the reconstruction
- Train using Backprop
- The code layer is a summary of the input
  - Somewhat similar to the alphas in PCA

Generative Adversarial Networks

- Combine several ideas:
  - Take a *generative* approach: the model can generate new fake data
  - Make it possible to adjust the parameters of the model with gradient descent
  - Use a deep network to model data hierarchically
  - **New:** learn two networks, a generator and a discriminator
    - The generator generates plausible fake samples
    - The discriminator tries to distinguish between different sample