CPSC 540: Machine Learning

Kernel Density Estimation, Factor Analysis

Mark Schmidt

University of British Columbia

Winter 2017

Admin

- Assignment 2:
  - 2 late days to hand it in tonight.
- Assignment 3:
  - Due Feburary 27.

Comments on TensorFlow Talk

- Most of the talk focused on large-scale issues, which I won't cover:
  - Synchronous vs. asynchronous (540 course project topic in 2014).
  - Issues related to distributed data/parameters.
- Some models were mentioned that I'm plannning to get to:
  - Word2vec.
  - RNNs.
  - LSTMs.
  - Sequence-to-sequence.
  - Neural machine translation.

Last Time: Mixture of Gaussians

- The classic mixture of Gaussians model uses a PDF of the form

$$p(x^i|\Theta) = \sum_{c=1}^{k} \underbrace{p(z^i = c|\Theta)}_{\text{prob(cluster)}} \underbrace{p(x^i|z^i = c, \Theta)}_{\text{prob}(x)\text{ given cluster}},$$

Last Time: Mixture of Gaussians

- The classic mixture of Gaussians model uses a PDF of the form

$$p(x^i|\Theta) = \sum_{c=1}^{k} \underbrace{p(z^i = c|\Theta)}_{\text{prob(cluster)}} \underbrace{p(x^i|z^i = c, \Theta)}_{\text{prob}(x)\text{ given cluster}},$$

where each mixture component is a multivariate Gaussian,

$$p(x^i|z^i = c, \Theta) = \frac{1}{(2\pi)^{\frac{d}{2}}|\Sigma_c|^{\frac{1}{2}}} \exp\left(-\frac{1}{2}(x^i - \mu_c)^T \Sigma_c^{-1}(x^i - \mu_c)\right),$$

Last Time: Mixture of Gaussians

- The classic mixture of Gaussians model uses a PDF of the form

$$p(x^i|\Theta) = \sum_{c=1}^{k} \underbrace{p(z^i = c|\Theta)}_{\text{prob(cluster)}} \underbrace{p(x^i|z^i = c, \Theta)}_{\text{prob}(x)\text{ given cluster}},$$

where each mixture component is a multivariate Gaussian,

$$p(x^i|z^i = c, \Theta) = \frac{1}{(2\pi)^{\frac{d}{2}}|\Sigma_c|^{\frac{1}{2}}} \exp\left(-\frac{1}{2}(x^i - \mu_c)^T\Sigma_c^{-1}(x^i - \mu_c)\right),$$

and we model the mixture probabilities as categorical,

$$p(z^i = c|\Theta) = \pi_c.$$

Last Time: Mixture of Gaussians

- The classic mixture of Gaussians model uses a PDF of the form

$$p(x^i|\Theta) = \sum_{c=1}^{k} \underbrace{p(z^i = c|\Theta)}_{\text{prob(cluster)}} \underbrace{p(x^i|z^i = c, \Theta)}_{\text{prob}(x)\text{ given cluster}},$$

where each mixture component is a multivariate Gaussian,

$$p(x^i|z^i = c, \Theta) = \frac{1}{(2\pi)^{\frac{d}{2}}|\Sigma_c|^{\frac{1}{2}}} \exp\left(-\frac{1}{2}(x^i - \mu_c)^T\Sigma_c^{-1}(x^i - \mu_c)\right),$$

and we model the mixture probabilities as categorical,

$$p(z^i = c|\Theta) = \pi_c.$$

- Finding the optimal parameter $\Theta = \{\pi_c, \mu_c, \Sigma_c\}_{c=1}^k$ is NP-hard.
  - But EM updates for improving parameters use analytic form of Gaussian MLE.

Expectation Maximization for Mixture of Gaussians

- EM update for mixture models is often written in terms of responsibilitites,

$$r_c^i \triangleq p(z^i = c|x^i, \Theta^t) = \frac{p(x^i|z^i = c, \Theta^t)p(z^i = c, \Theta^t)}{\sum_{c'=1}^{k} p(x^i|z^i = c', \Theta^t)p(z^i = c', \Theta^t)},$$

a number between 0 and 1 giving probability that cluster c generated x^i.

Expectation Maximization for Mixture of Gaussians

- EM update for mixture models is often written in terms of responsibilitites,

$$r_c^i \triangleq p(z^i = c | x^i, \Theta^t) = \frac{p(x^i | z^i = c, \Theta^t) p(z^i = c, \Theta^t)}{\sum_{c'=1}^{k} p(x^i | z^i = c', \Theta^t) p(z^i = c', \Theta^t)},$$

  a number between 0 and 1 giving probability that cluster c generated x^i.
- For mixture of Gaussian, EM updates takes the form

$$\pi_c^{t+1} = \frac{1}{n} \sum_{i=1}^{n} r_c^i \qquad \text{(proportion of examples soft-assigned to cluster } c\text{)}$$

$$\mu_c^{t+1} = \frac{\sum_{i=1}^{n} r_c^i x^i}{n \pi_c^{t+1}} \qquad \text{(mean of examples soft-assigned to cluster } c\text{)}$$

$$\Sigma_c^{t+1} = \frac{\sum_{i=1}^{n} r_c^i (x^i - \mu_c^{t+1})(x^i - \mu_c^{t+1})^T}{n \pi_c^{t+1}} \qquad \text{(covariance of examples soft-assigned to } c\text{).}$$

Expectation Maximization for Mixture of Gaussians

- EM update for mixture models is often written in terms of responsibilitites,

$$r_c^i \triangleq p(z^i = c | x^i, \Theta^t) = \frac{p(x^i | z^i = c, \Theta^t) p(z^i = c, \Theta^t)}{\sum_{c'=1}^k p(x^i | z^i = c', \Theta^t) p(z^i = c', \Theta^t)},$$

  a number between 0 and 1 giving probability that cluster c generated x^i.
- For mixture of Gaussian, EM updates takes the form

$$\pi_c^{t+1} = \frac{1}{n}\sum_{i=1}^n r_c^i \quad \text{(proportion of examples soft-assigned to cluster } c\text{)}$$

$$\mu_c^{t+1} = \frac{\sum_{i=1}^n r_c^i x^i}{n\pi_c^{t+1}} \quad \text{(mean of examples soft-assigned to cluster } c\text{)}$$

$$\Sigma_c^{t+1} = \frac{\sum_{i=1}^n r_c^i (x^i - \mu_c^{t+1})(x^i - \mu_c^{t+1})^T}{n\pi_c^{t+1}} \quad \text{(covariance of examples soft-assigned to } c\text{).}$$

- Derivation is tedious (see notes on webpage).
  - Uses distributive law, probabilities sum to one, Lagrangian, weighted Gaussian MLE.
- We get k-means if $r_c^i = 1$ for most likely cluster, and Σ_c is constant across c.

Expectation Maximization for Mixture of Gaussians

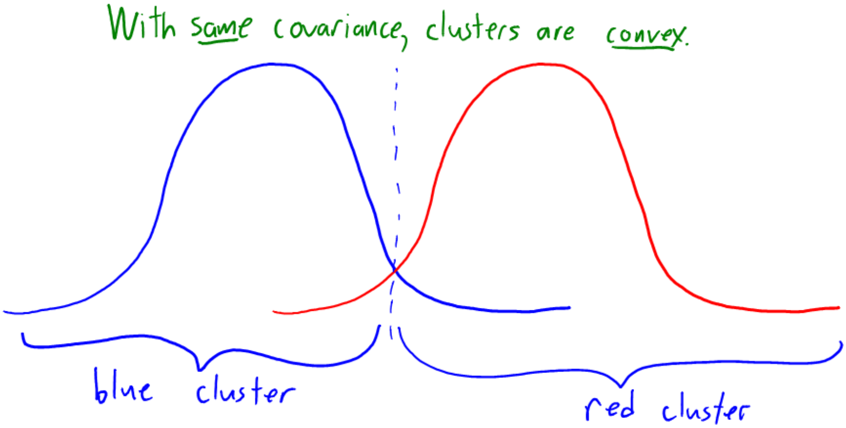
- EM for fitting mixture of Gaussians in action:
  `https://www.youtube.com/watch?v=B36fzChfyGU`

K-Means vs. Mixture of Gaussians

- K-means is special case of “hard EM” for mixture of Gaussians (common Σ_c).
  - But EM allows points to be assigned to be multiple clusters

K-Means vs. Mixture of Gaussians

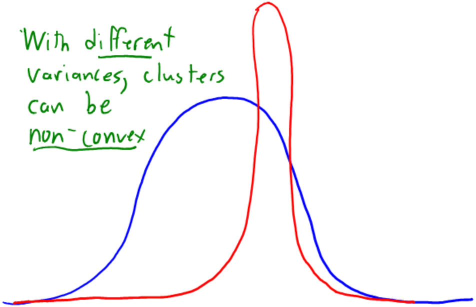
- K-means is special case of "hard EM" for mixture of Gaussians (common Σ_c).
  - But EM allows points to be assigned to be multiple clusters
  - General Σ_c in mixture of Gaussians allow non-convex clusters.



K-Means vs. Mixture of Gaussians

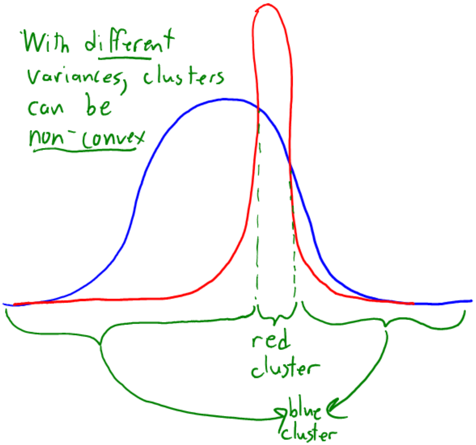
- K-means is special case of "hard EM" for mixture of Gaussians (common Σ_c).
  - But EM allows points to be assigned to be multiple clusters
  - General Σ_c in mixture of Gaussians allow non-convex clusters.

With different variances, clusters can be non-convex



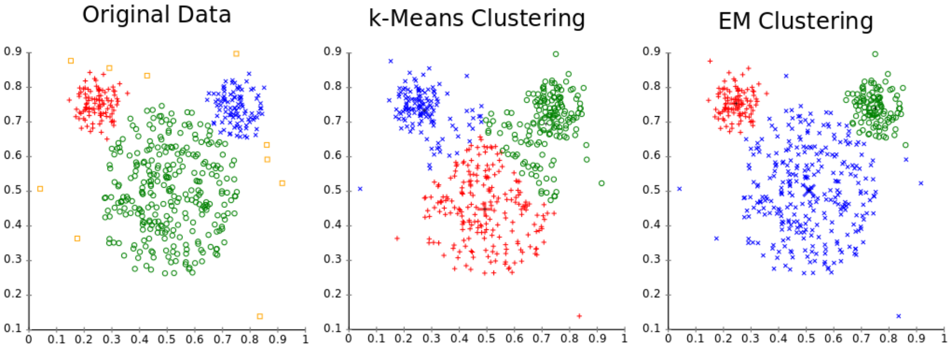
K-Means vs. Mixture of Gaussians

- K-means is special case of "hard EM" for mixture of Gaussians (common Σ_c).
  - But EM allows points to be assigned to be multiple clusters
  - General Σ_c in mixture of Gaussians allow non-convex clusters.



K-Means vs. Mixture of Gaussians

- K-means is special case of "hard EM" for mixture of Gaussians (common Σ_c).
  - But EM allows points to be assigned to be multiple clusters
  - General Σ_c in mixture of Gaussians allow non-convex clusters.



https://en.wikipedia.org/wiki/K-means_clustering

Outline

1. Monotonicity of EM
2. Kernel Density Estimation
3. Factor Analysis

Expectation Maximization

- EM considers learning with observed variables O and hidden variables H.
- In this case the "observed" marginal log-liklihooed has a nasty form,

$$\log p(O|\Theta) = \log\left(\sum_H p(O,H|\Theta)\right).$$

Expectation Maximization

- EM considers learning with observed variables O and hidden variables H.
- In this case the "observed" marginal log-liklihooed has a nasty form,

$$\log p(O|\Theta) = \log\left(\sum_H p(O,H|\Theta)\right).$$

- EM applies when "complete" likelihood, $p(O,H|\Theta)$, has a nice form.
- EM iterations take the form

$$\Theta^{t+1} = \underset{\Theta}{\text{argmax}}\left\{\sum_H \alpha_H \log p(O,H|\Theta)\right\},$$

where $\alpha_H = p(H|O,\Theta^t)$.

Bound on Progress of Expectation Maximization

The iterations of the EM algorithm satisfy

$$\log p(O|\Theta^{t+1}) - \log p(O|\Theta^t) \geq Q(\Theta^{t+1}|\Theta^t) - Q(\Theta^t|\Theta^t),$$

Bound on Progress of Expectation Maximization

The iterations of the EM algorithm satisfy

$$\log p(O|\Theta^{t+1}) - \log p(O|\Theta^{t}) \geq Q(\Theta^{t+1}|\Theta^{t}) - Q(\Theta^{t}|\Theta^{t}),$$

- Proof:

$$\begin{aligned} -\log p(O|\Theta) &= -\log\left(\sum_h p(O,H|\Theta)\right) \\ &= -\log\left(\sum_H \alpha_H \frac{p(O,H|\Theta)}{\alpha_H}\right) \qquad \text{(for } \alpha_H \neq 0\text{)} \end{aligned}$$

Bound on Progress of Expectation Maximization

The iterations of the EM algorithm satisfy

$$\log p(O|\Theta^{t+1}) - \log p(O|\Theta^{t}) \geq Q(\Theta^{t+1}|\Theta^{t}) - Q(\Theta^{t}|\Theta^{t}),$$

- Proof:

$$\begin{aligned} -\log p(O|\Theta) &= -\log\left(\sum_h p(O,H|\Theta)\right) \\ &= -\log\left(\sum_H \alpha_H \frac{p(O,H|\Theta)}{\alpha_H}\right) \qquad \text{(for } \alpha_H \neq 0\text{)} \\ &\leq -\sum_H \alpha_H \log\left(\frac{p(O,H|\Theta)}{\alpha_H}\right), \end{aligned}$$

becuase $-\log(z)$ is convex.

Bound on Progress of Expectation Maximization

The iterations of the EM algorithm satisfy

$$\log p(O|\Theta^{t+1}) - \log p(O|\Theta^t) \geq Q(\Theta^{t+1}|\Theta^t) - Q(\Theta^t|\Theta^t),$$

- Using that log turns multiplication into addition we get

$$\begin{aligned}
-\log p(O|\Theta) &\leq -\sum_H \alpha_H \log\left(\frac{p(O,H|\Theta)}{\alpha_H}\right)\\
&= \underbrace{-\sum_H \alpha_H \log p(O,H|\Theta)}_{Q(\Theta|\Theta^t)} + \underbrace{\sum_H \alpha_H \log \alpha_H}_{\text{negative entropy}}
\end{aligned}$$

Bound on Progress of Expectation Maximization

The iterations of the EM algorithm satisfy

$$\log p(O|\Theta^{t+1}) - \log p(O|\Theta^t) \geq Q(\Theta^{t+1}|\Theta^t) - Q(\Theta^t|\Theta^t),$$

- Using that log turns multiplication into addition we get

$$\begin{aligned}
-\log p(O|\Theta) &\leq -\sum_H \alpha_H \log\left(\frac{p(O,H|\Theta)}{\alpha_H}\right) \\
&= \underbrace{-\sum_H \alpha_H \log p(O,H|\Theta)}_{Q(\Theta|\Theta^t)} + \underbrace{\sum_H \alpha_H \log \alpha_H}_{\text{negative entropy}} \\
&= -Q(\Theta|\Theta^t) - \text{entropy}(\alpha),
\end{aligned}$$

which we can use to bound $\log p(O|\Theta^{t+1})$.

Bound on Progress of Expectation Maximization

The iterations of the EM algorithm satisfy

$$\log p(O|\Theta^{t+1}) - \log p(O|\Theta^t) \geq Q(\Theta^{t+1}|\Theta^t) - Q(\Theta^t|\Theta^t),$$

- To bound $p(O|\Theta^t)$ we use definition of conditional,

$$p(H|O,\Theta^t) = \frac{p(O,H|\Theta^t)}{p(O|\Theta^t)}$$

Bound on Progress of Expectation Maximization

The iterations of the EM algorithm satisfy

$$\log p(O|\Theta^{t+1}) - \log p(O|\Theta^t) \geq Q(\Theta^{t+1}|\Theta^t) - Q(\Theta^t|\Theta^t),$$

- To bound $p(O|\Theta^t)$ we use definition of conditional,

$$p(H|O,\Theta^t) = \frac{p(O,H|\Theta^t)}{p(O|\Theta^t)} \quad \text{or} \quad \log p(O|\Theta^t) = \log p(O,H|\Theta^t) - \log p(H|O,\Theta^t).$$

Bound on Progress of Expectation Maximization

The iterations of the EM algorithm satisfy

$$\log p(O|\Theta^{t+1}) - \log p(O|\Theta^t) \geq Q(\Theta^{t+1}|\Theta^t) - Q(\Theta^t|\Theta^t),$$

- To bound $p(O|\Theta^t)$ we use definition of conditional,

$$p(H|O,\Theta^t) = \frac{p(O,H|\Theta^t)}{p(O|\Theta^t)} \quad \text{or} \quad \log p(O|\Theta^t) = \log p(O,H|\Theta^t) - \log p(H|O,\Theta^t).$$

- Multiply by α_H and sum equality over H values,

$$\sum_H \alpha_H \log p(O|\Theta^t) = \sum_H \alpha_H \log p(O,H|\Theta^t) - \sum_H \alpha_H \log p(H|O,\Theta^t).$$

Bound on Progress of Expectation Maximization

The iterations of the EM algorithm satisfy

$$\log p(O|\Theta^{t+1}) - \log p(O|\Theta^t) \geq Q(\Theta^{t+1}|\Theta^t) - Q(\Theta^t|\Theta^t),$$

- To bound $p(O|\Theta^t)$ we use definition of conditional,

$$p(H|O,\Theta^t) = \frac{p(O,H|\Theta^t)}{p(O|\Theta^t)} \quad \text{or} \quad \log p(O|\Theta^t) = \log p(O,H|\Theta^t) - \log p(H|O,\Theta^t).$$

- Multiply by α_H and sum equality over H values,

$$\sum_H \alpha_H \log p(O|\Theta^t) = \sum_H \alpha_H \log p(O,H|\Theta^t) - \sum_H \alpha_H \log p(H|O,\Theta^t).$$

- Using the EM definition of α_h we have

$$\log p(O|\Theta^t) \underbrace{\sum_H \alpha_H}_{=1} = Q(\Theta^t|\Theta^t) + \text{entropy}(\alpha).$$

Bound on Progress of Expectation Maximization

The iterations of the EM algorithm satisfy

$$\log p(O|\Theta^{t+1}) - \log p(O|\Theta^t) \geq Q(\Theta^{t+1}|\Theta^t) - Q(\Theta^t|\Theta^t),$$

- Thus we have the two bounds

$$\log p(O|\Theta) \geq Q(\Theta|\Theta^t) + \text{entropy}(\alpha)$$
$$\log p(O|\Theta^t) = Q(\Theta^t|\Theta^t) + \text{entropy}(\alpha).$$

Subtracting these and using $\Theta = \Theta^{t+1}$ gives the result.

Bound on Progress of Expectation Maximization

The iterations of the EM algorithm satisfy

$$\log p(O|\Theta^{t+1}) - \log p(O|\Theta^t) \geq Q(\Theta^{t+1}|\Theta^t) - Q(\Theta^t|\Theta^t),$$

- Thus we have the two bounds

$$\log p(O|\Theta) \geq Q(\Theta|\Theta^t) + \text{entropy}(\alpha)$$
$$\log p(O|\Theta^t) = Q(\Theta^t|\Theta^t) + \text{entropy}(\alpha).$$

  Subtracting these and using $\Theta = \Theta^{t+1}$ gives the result.

- Inequality holds for any choice of Θ^{t+1}.
  - Approximate M-steps are ok: we just need to decrease Q to improve likelihood.
- Implies entropy of α_H gives tightness of bound.
  - If variables are "predictable" then the bound is tight and we get "hard" EM.

Discussing of EM for Mixtures of Gaussians

- EM and mixture models are used in a ton of applications.
  - One of the default unsupervised learning methods.

Discussing of EM for Mixtures of Gaussians

- EM and mixture models are used in a ton of applications.
  - One of the default unsupervised learning methods.
- EM usually doesn't reach global optimum.
  - Classic solution: restart the algorithm from different initializations.

Discussing of EM for Mixtures of Gaussians

- EM and mixture models are used in a ton of applications.
  - One of the default unsupervised learning methods.
- EM usually doesn't reach global optimum.
  - Classic solution: restart the algorithm from different initializations.
- MLE for some clusters may not exist (e.g., only responsible for one point).
  - Use MAP estimates or remove these clusters.

Discussing of EM for Mixtures of Gaussians

- EM and mixture models are used in a ton of applications.
  - One of the default unsupervised learning methods.
- EM usually doesn't reach global optimum.
  - Classic solution: restart the algorithm from different initializations.
- MLE for some clusters may not exist (e.g., only responsible for one point).
  - Use MAP estimates or remove these clusters.
- How do you choose number of mixtures k?
  - Use cross-validation or other model selection criteria.

Discussing of EM for Mixtures of Gaussians

- EM and mixture models are used in a ton of applications.
  - One of the default unsupervised learning methods.
- EM usually doesn't reach global optimum.
  - Classic solution: restart the algorithm from different initializations.
- MLE for some clusters may not exist (e.g., only responsible for one point).
  - Use MAP estimates or remove these clusters.
- How do you choose number of mixtures k?
  - Use cross-validation or other model selection criteria.
- Can you make it robust?
  - Use mixture of Laplace of student t distributions.

Discussing of EM for Mixtures of Gaussians

- EM and mixture models are used in a ton of applications.
    - One of the default unsupervised learning methods.
- EM usually doesn't reach global optimum.
    - Classic solution: restart the algorithm from different initializations.
- MLE for some clusters may not exist (e.g., only responsible for one point).
    - Use MAP estimates or remove these clusters.
- How do you choose number of mixtures k?
    - Use cross-validation or other model selection criteria.
- Can you make it robust?
    - Use mixture of Laplace of student t distributions.
- Are there alternatives to EM?
    - Could use gradient descent on NLL.
    - Spectral and other recent methods have some global guarantees.

Outline

1. Monotonicity of EM
2. Kernel Density Estimation
3. Factor Analysis

A Non-Parametric Mixture Model

- The classic parametric mixture model has the form

$$p(x) = \sum_{c=1}^{k} p(z = c)p(x|z = c).$$

A Non-Parametric Mixture Model

- The classic parametric mixture model has the form

$$p(x) = \sum_{c=1}^{k} p(z = c)p(x|z = c).$$

- A natural way to define a non-parametric mixture model is

$$p(x) = \sum_{i=1}^{n} p(z = i)p(x|z = i),$$

  where we have one mixture for every training example i.

A Non-Parametric Mixture Model

- The classic parametric mixture model has the form

$$p(x) = \sum_{c=1}^{k} p(z = c)p(x|z = c).$$

- A natural way to define a non-parametric mixture model is

$$p(x) = \sum_{i=1}^{n} p(z = i)p(x|z = i),$$

  where we have one mixture for every training example i.
- Common example: z is uniform and $x|z$ is Gaussian with mean x^i,

$$p(x) = \frac{1}{n}\sum_{i=1}^{n} \mathcal{N}(x|x^i, \sigma^2 I),$$

  and we use a shared covariance $\sigma^2 I$ (with σ estimated by cross-validation).

A Non-Parametric Mixture Model

- The classic parametric mixture model has the form

$$p(x) = \sum_{c=1}^{k} p(z = c)p(x|z = c).$$

- A natural way to define a non-parametric mixture model is

$$p(x) = \sum_{i=1}^{n} p(z = i)p(x|z = i),$$

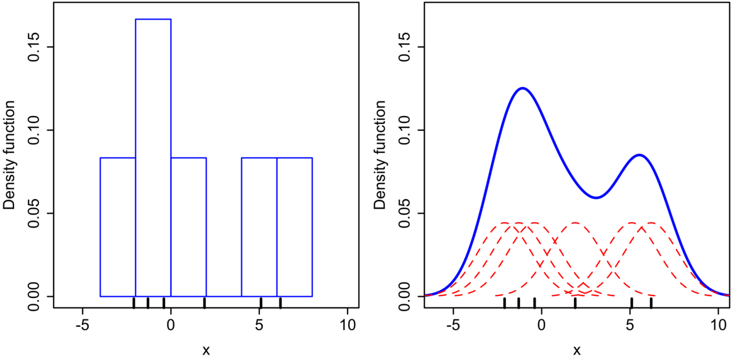
  where we have one mixture for every training example i.
- Common example: z is uniform and $x|z$ is Gaussian with mean x^i,

$$p(x) = \frac{1}{n}\sum_{i=1}^{n} \mathcal{N}(x|x^i, \sigma^2 I),$$

  and we use a shared covariance $\sigma^2 I$ (with σ estimated by cross-validation).
- This is a special case of kernel density estimation or Parzen window.

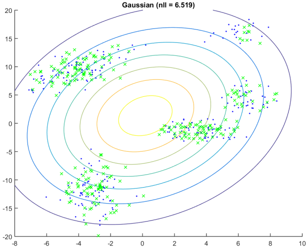
Histogram vs. Kernel Density Estimator

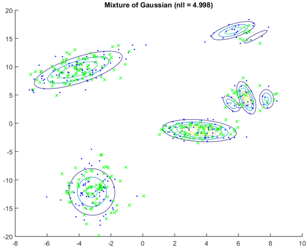
- Think of kernel density estimator as a smooth version of histogram:



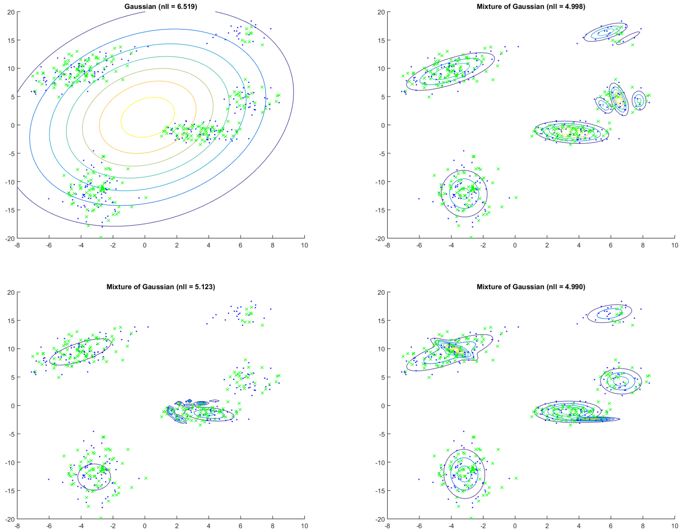
https://en.wikipedia.org/wiki/Kernel_density_estimation

Parzen Window vs. Gaussian and Mixture of Gaussian

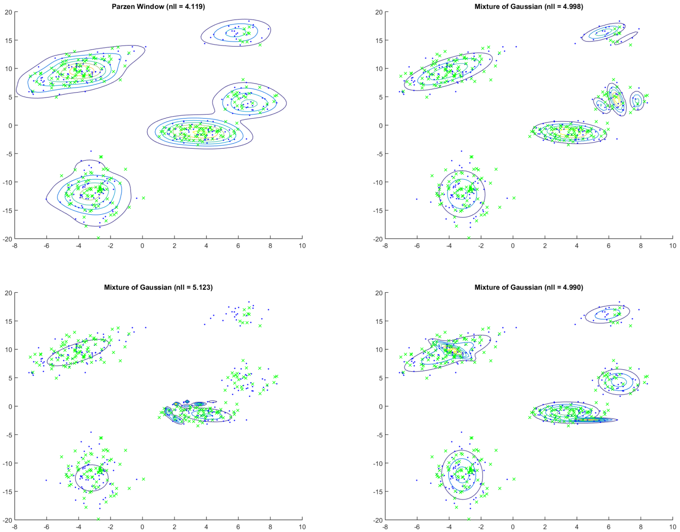




Parzen Window vs. Gaussian and Mixture of Gaussian



Parzen Window vs. Gaussian and Mixture of Gaussian



Kernel Density Estimation

- The 1D kernel density estimation (KDE) model uses

$$p(x) = \frac{1}{n}\sum_{i=1}^{n} k_\sigma(x - x^i),$$

where the PDF k is the "kernel" and the parameter σ is the "bandwidth".

Kernel Density Estimation

- The 1D kernel density estimation (KDE) model uses

$$p(x) = \frac{1}{n}\sum_{i=1}^{n} k_\sigma(x - x^i),$$

  where the PDF k is the "kernel" and the parameter σ is the "bandwidth".
- In the previous slide we used the (normalized) Gaussian kernel,

$$k_1(x) = \frac{1}{\sqrt{2\pi}}\exp\left(-\frac{x^2}{2}\right), \quad k_\sigma(x) = \frac{1}{\sigma\sqrt{2\pi}}\exp\left(-\frac{x^2}{2\sigma^2}\right).$$

Kernel Density Estimation

- The 1D kernel density estimation (KDE) model uses

$$p(x) = \frac{1}{n}\sum_{i=1}^{n} k_\sigma(x - x^i),$$

where the PDF k is the "kernel" and the parameter σ is the "bandwidth".

- In the previous slide we used the (normalized) Gaussian kernel,

$$k_1(x) = \frac{1}{\sqrt{2\pi}}\exp\left(-\frac{x^2}{2}\right), \quad k_\sigma(x) = \frac{1}{\sigma\sqrt{2\pi}}\exp\left(-\frac{x^2}{2\sigma^2}\right).$$

- Note that we can add a bandwith σ to any PDF k_1, using

$$k_\sigma(x) = \frac{1}{\sigma}k_1\left(\frac{x}{\sigma}\right),$$

which follows from the change of variables formula for probabilities.

- Under common choices of kernels, KDEs can model any continuous density.

Efficient Kernel Density Estimation

- KDE with the Gaussian kernel is slow at test time:
  - We need to compute distance of test point to every training point.

Efficient Kernel Density Estimation

- KDE with the Gaussian kernel is slow at test time:
  - We need to compute distance of test point to every training point.
- A common alternative is the Epanechnikov kernel,

$$k_1(x) = \frac{3}{4}\left(1 - x^2\right)\mathcal{I}\left[|x| \leq 1\right].$$

- This kernel has two nice properties:
  - Epanechnikov showed that it is asymptotically optimal in terms of squared error.
  - It can be much faster to use since it only depends on nearby points.
    - You can use fast methods for computing nearest neighbours.

Efficient Kernel Density Estimation

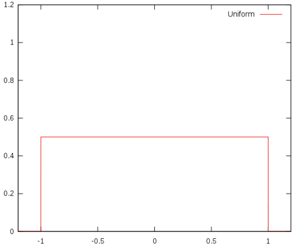
- KDE with the Gaussian kernel is slow at test time:
  - We need to compute distance of test point to every training point.
- A common alternative is the Epanechnikov kernel,

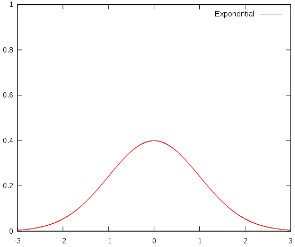
$$k_1(x) = \frac{3}{4}\left(1 - x^2\right)\mathcal{I}\left[|x| \leq 1\right].$$

- This kernel has two nice properties:
  - Epanechnikov showed that it is asymptotically optimal in terms of squared error.
  - It can be much faster to use since it only depends on nearby points.
    - You can use fast methods for computing nearest neighbours.
- It is non-smooth at the boundaries but many smooth approximations exist.
  - Quartic, triweight, tricube, cosine, etc.

Visualization of Common Kernel Functions

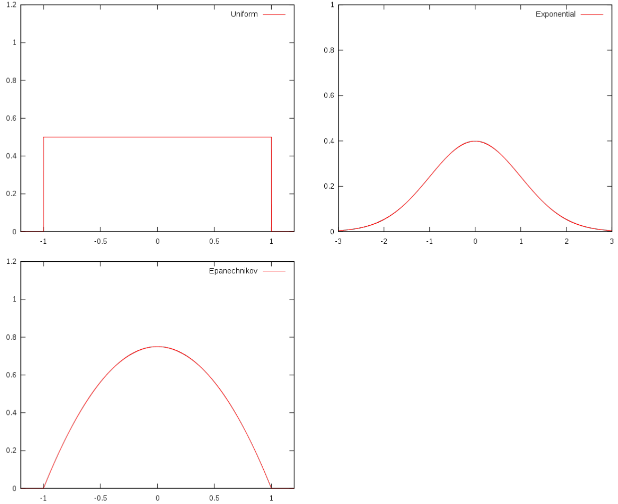
Histogram vs. Gaussian vs. Epanechnikov vs. tricube:





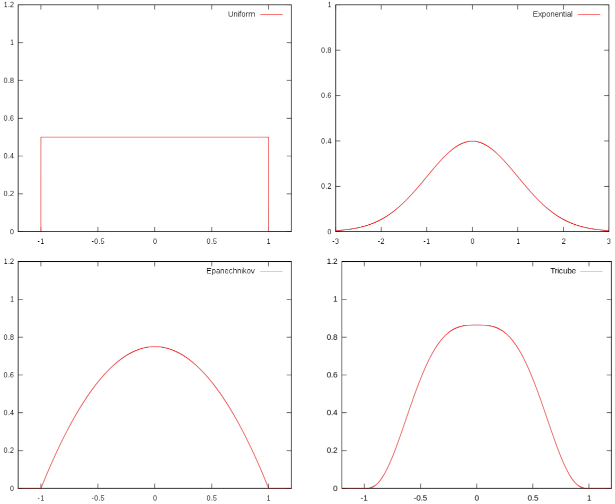
Visualization of Common Kernel Functions

Histogram vs. Gaussian vs. Epanechnikov vs. tricube:



Visualization of Common Kernel Functions

Histogram vs. Gaussian vs. Epanechnikov vs. tricube:



Multivariate Kernel Density Estimation

- The multivariate kernel density estimation (KDE) model uses

$$p(x) = \frac{1}{n}\sum_{i=1}^{n} k_R(x - x^i),$$

- The most common kernel is again the Gaussian,

$$k_I(x) = \frac{1}{\sqrt{2\pi}^{\frac{d}{2}}} \exp\left(-\frac{\|x\|^2}{2}\right).$$

Multivariate Kernel Density Estimation

- The multivariate kernel density estimation (KDE) model uses

$$p(x) = \frac{1}{n}\sum_{i=1}^{n} k_R(x - x^i),$$

- The most common kernel is again the Gaussian,

$$k_I(x) = \frac{1}{\sqrt{2\pi}^{\frac{d}{2}}} \exp\left(-\frac{\|x\|^2}{2}\right).$$

- We can add a bandwith matrix R to any kernel using

$$k_R(x) = \frac{1}{|R|} k_1(R^{-1}x) \qquad \text{(generalizes } k_\sigma(x) = \frac{1}{\sigma} k_1\left(\frac{x}{\sigma}\right)\text{)},$$

  and multivariate Gaussian with covariance Σ corresponds to $R = \Sigma^{\frac{1}{2}}$.
- To reduce number of parameters, we typically:
  - Use a product of independent distributions and use $R = \sigma I$ for some σ.

Mean-Shift Clustering

- Mean-shift clustering uses KDE for clustering:
  - Fit a KDE to the training examples, and then for test example $\hat{x}$:
    - Run gradient descent starting from $\hat{x}$.
  - Clusters are points that reach same local minimum.
- `https://spin.atomicobject.com/2015/05/26/mean-shift-clustering`

Mean-Shift Clustering

- Mean-shift clustering uses KDE for clustering:
  - Fit a KDE to the training examples, and then for test example $\hat{x}$:
    - Run gradient descent starting from $\hat{x}$.
  - Clusters are points that reach same local minimum.
- https://spin.atomicobject.com/2015/05/26/mean-shift-clustering
- Not sensitive to initialization, no need to choose k, can find non-convex clusters.
- Similar to density-based clustering from 340.
  - But doesn't require uniform density within cluster.
  - And can be used for vector quantization.

Outline

Expectation Maximization with Many Discrete Variables

- EM iterations take the form

$$\Theta^{t+1} = \underset{\Theta}{\operatorname{argmax}} \left\{ \sum_{H} \alpha_H \log p(O, H|\Theta) \right\},$$

and with multiple MAR variables $\{H_1, H_2, \ldots, H_m\}$ this means

$$\Theta^{t+1} = \underset{\Theta}{\operatorname{argmax}} \left\{ \sum_{H_1} \sum_{H_2} \cdots \sum_{H_m} \alpha_H \log p(O, H|\Theta) \right\},$$

Expectation Maximization with Many Discrete Variables

- EM iterations take the form

$$\Theta^{t+1} = \underset{\Theta}{\text{argmax}} \left\{ \sum_H \alpha_H \log p(O, H|\Theta) \right\},$$

and with multiple MAR variables $\{H_1, H_2, \dots, H_m\}$ this means

$$\Theta^{t+1} = \underset{\Theta}{\text{argmax}} \left\{ \sum_{H_1} \sum_{H_2} \cdots \sum_{H_m} \alpha_H \log p(O, H|\Theta) \right\},$$

- In mixture models, EM sums over all 2^n possible cluster assignments.
- In binary semi-supervised learning, EM sums over all 2^t assignments to $\tilde{y}$.

Expectation Maximization with Many Discrete Variables

- EM iterations take the form

$$\Theta^{t+1} = \operatorname*{argmax}_{\Theta} \left\{ \sum_{H} \alpha_H \log p(O, H | \Theta) \right\},$$

  and with multiple MAR variables $\{H_1, H_2, \dots, H_m\}$ this means

$$\Theta^{t+1} = \operatorname*{argmax}_{\Theta} \left\{ \sum_{H_1} \sum_{H_2} \cdots \sum_{H_m} \alpha_H \log p(O, H | \Theta) \right\},$$

- In mixture models, EM sums over all 2^n possible cluster assignments.
- In binary semi-supervised learning, EM sums over all 2^t assignments to $\tilde{y}$.

- But conditional independence allows efficient calculation in the above cases.
  - The H are independent given $\{O, \Theta\}$ which simplifies sums (see EM notes).
  - We'll cover general case when we discuss probabilistic graphical models.

Today: Continuous-Latent Variables

- If H is continuous, the sums are replaceed by integrals,

$$\log p(O|\Theta) = \log\left(\int_H p(O, H|\Theta)dH\right) \qquad \text{(log-likelihood)}$$

$$\Theta^{t+1} = \underset{\Theta}{\text{argmax}}\left\{\int_H \alpha_H \log p(O, H|\Theta)dH\right\} \qquad \text{(EM update)},$$

  where if have 5 hidden varialbes $\int_H$ means $\int_{H_1}\int_{H_2}\int_{H_3}\int_{H_4}\int_{H_5}$.
- Even with conditional independence these might be hard.

Today: Continuous-Latent Variables

- If H is continuous, the sums are replaceed by integrals,

$$\log p(O|\Theta) = \log\left(\int_H p(O,H|\Theta)dH\right) \qquad \text{(log-likelihood)}$$

$$\Theta^{t+1} = \underset{\Theta}{\text{argmax}}\left\{\int_H \alpha_H \log p(O,H|\Theta)dH\right\} \qquad \text{(EM update)},$$

  where if have 5 hidden varialbes $\int_H$ means $\int_{H_1}\int_{H_2}\int_{H_3}\int_{H_4}\int_{H_5}$.
- Even with conditional independence these might be hard.

- Gaussian assumptions allow efficient calculation of these integrals.
  - We'll cover general case when we get discuss Bayesian statistics.

Today: Continuous-Latent Variables

- In mixture models, we have a discrete latent variable z:
  - In mixture of Gaussians, if you know the cluster z then $p(x|z)$ is a Gaussian.

Today: Continuous-Latent Variables

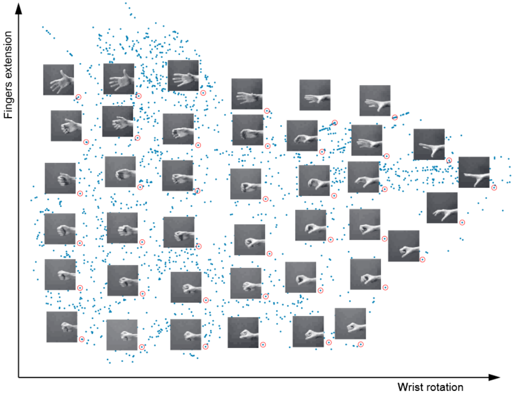
- In mixture models, we have a discrete latent variable z:
  - In mixture of Gaussians, if you know the cluster z then $p(x|z)$ is a Gaussian.
- In latent-factor models, we have continuous latent variables z:
  - In probabilistic PCA, if you know the latent-factors z then $p(x|z)$ is a Gaussian.

Today: Continuous-Latent Variables

- In mixture models, we have a discrete latent variable z:
  - In mixture of Gaussians, if you know the cluster z then $p(x|z)$ is a Gaussian.
- In latent-factor models, we have continuous latent variables z:
  - In probabilistic PCA, if you know the latent-factors z then $p(x|z)$ is a Gaussian.
- But what would a continuous z be useful for?
- Do we really need to start solving integrals?

Today: Continuous-Latent Variables

- Data may live in a low-dimensional manifold:



http://isomap.stanford.edu/handfig.html

- Mixtures are inefficient at representing the 2D manifold.

Principal Component Analysis (PCA)

- PCA replaces X with a lower-dimensional approximation Z.
    - Matrix Z has n rows, but typically far fewer columns.

Principal Component Analysis (PCA)

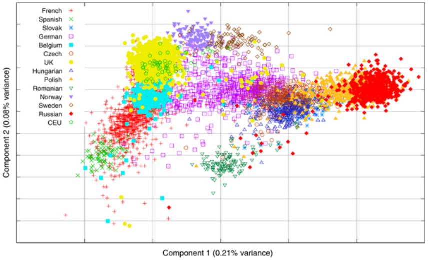
- PCA replaces X with a lower-dimensional approximation Z.
  - Matrix Z has n rows, but typically far fewer columns.
- PCA is used for:
  - Dimensionality reduction: replace X with a lower-dimensional Z.

Principal Component Analysis (PCA)

- PCA replaces X with a lower-dimensional approximation Z.
  - Matrix Z has n rows, but typically far fewer columns.
- PCA is used for:
  - Dimensionality reduction: replace X with a lower-dimensional Z.
  - Outlier detection: if PCA gives poor approximation of x^i, could be outlier.
  - Basis for linear models: use Z as features in regression model.

Principal Component Analysis (PCA)

- PCA replaces X with a lower-dimensional approximation Z.
    - Matrix Z has n rows, but typically far fewer columns.
- PCA is used for:
    - Dimensionality reduction: replace X with a lower-dimensional Z.
    - Outlier detection: if PCA gives poor approximation of x^i, could be outlier.
    - Basis for linear models: use Z as features in regression model.
    - Data visualization: display z^i in a scatterplot.
    - Factor discovering: discover important hidden "factors" underlying data.



http://infoproc.blogspot.ca/2008/11/european-genetic-substructure.html

PCA Notation

- PCA approximates the original matrix by factor-loadings Z and latent-factors W,

$$X \approx ZW.$$

  where $Z \in \mathbb{R}^{n \times k}$, $W \in \mathbb{R}^{k \times d}$, and we assume columns of X have mean 0.
- We're trying to split redundancy in X into its important "parts".

PCA Notation

- PCA approximates the original matrix by factor-loadings Z and latent-factors W,

$$X \approx ZW.$$

  where $Z \in \mathbb{R}^{n\times k}$, $W \in \mathbb{R}^{k\times d}$, and we assume columns of X have mean 0.
- We're trying to split redundancy in X into its important "parts".
- We typically take $k << d$ so this requires far fewer parameters:

$$\underbrace{\begin{bmatrix} \quad\quad\quad\quad & \\ \\ \\ \\ \end{bmatrix}}_{X\in\mathbb{R}^{n\times d}} \approx \underbrace{\begin{bmatrix} \quad \\ \\ \\ \\ \end{bmatrix}}_{Z\in\mathbb{R}^{n\times k}} \underbrace{\begin{bmatrix} \quad\quad\quad\quad & \\ \\ \end{bmatrix}}_{W\in\mathbb{R}^{k\times d}}$$

- Also computationally convenient:
  - Xv costs $O(nd)$ but $Z(Wv)$ only costs $O(nk + dk)$.

PCA Notation

- Using $X \approx ZW$, PCA approximates each examples x^i as

$$x^i \approx W^T z^i.$$

PCA Notation

- Using $X \approx ZW$, PCA approximates each examples x^i as

$$x^i \approx W^T z^i.$$

- Usually we only need to estimate W:
  - If using least squares, then given W we can find z^i from x^i using

$$z^i = \underset{z}{\operatorname{argmin}} \|x^i - W^T z\|^2 = (WW^T)^{-1} W x^i.$$

PCA Notation

- Using $X \approx ZW$, PCA approximates each examples x^i as

$$x^i \approx W^T z^i.$$

- Usually we only need to estimate W:
  - If using least squares, then given W we can find z^i from x^i using

$$z^i = \underset{z}{\operatorname{argmin}} \|x^i - W^T z\|^2 = (WW^T)^{-1} W x^i.$$

- We often assume that W^T is orthogonal:
  - This means that $WW^T = I$.
  - In this case we have $z^i = Wx^i$.
- In standard formulations, solution only unique up to rotation:
  - Usually, we fit the rows of W sequentially for uniqueness.

Two Classic Views on PCA

- PCA approximates the original matrix by latent-variables Z and latent-factors W,

$$X \approx ZW.$$

  where $Z \in \mathbb{R}^{n\times k}$, $W \in \mathbb{R}^{k\times d}$.
- Two classical interpretations/derivations of PCA:

Two Classic Views on PCA

- PCA approximates the original matrix by latent-variables Z and latent-factors W,

$$X \approx ZW.$$

  where $Z \in \mathbb{R}^{n \times k}$, $W \in \mathbb{R}^{k \times d}$.
- Two classical interpretations/derivations of PCA:
  1. Choose latent-factors W to minimize error ("synthesis view"):

$$\underset{Z \in \mathbb{R}^{n \times k}, W \in \mathbb{R}^{k \times d}}{\text{argmin}} \|X - ZW\|_F^2 = \sum_{i=1}^{n} \sum_{j=1}^{d} (x_j^i - (w_j)^T z^i)^2.$$

Two Classic Views on PCA

- PCA approximates the original matrix by latent-variables Z and latent-factors W,

$$X \approx ZW.$$

  where $Z \in \mathbb{R}^{n \times k}$, $W \in \mathbb{R}^{k \times d}$.

- Two classical interpretations/derivations of PCA:
  1. Choose latent-factors W to minimize error ("synthesis view"):

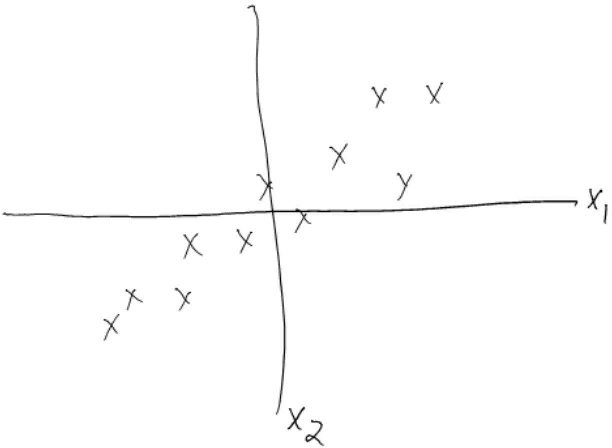
$$\underset{Z \in \mathbb{R}^{n \times k}, W \in \mathbb{R}^{k \times d}}{\text{argmin}} \|X - ZW\|_F^2 = \sum_{i=1}^{n} \sum_{j=1}^{d} (x_j^i - (w_j)^T z^i)^2.$$

  2. Choose orthogonal latent-factors W to maximize variance ("analysis view"):

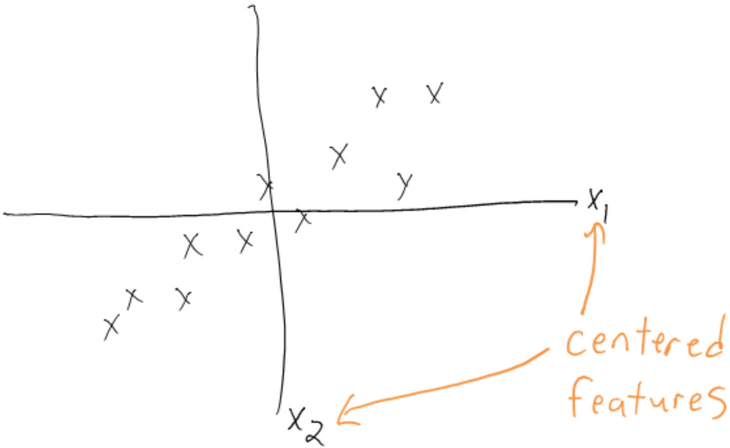
$$\underset{W \in \mathbb{R}^{k \times d}}{\text{argmax}} = \sum_{i=1}^{n} \|z^i - \mu_z\|^2 = \sum_{i=1}^{n} \|Wx^i\|^2 \qquad (z^i = Wx^i \text{ and } \mu_z = 0)$$

$$= \text{Tr}(WX^TXW^T) = n\text{Tr}(W^TWS) \qquad (\text{where } S \text{ is sample covariance})$$

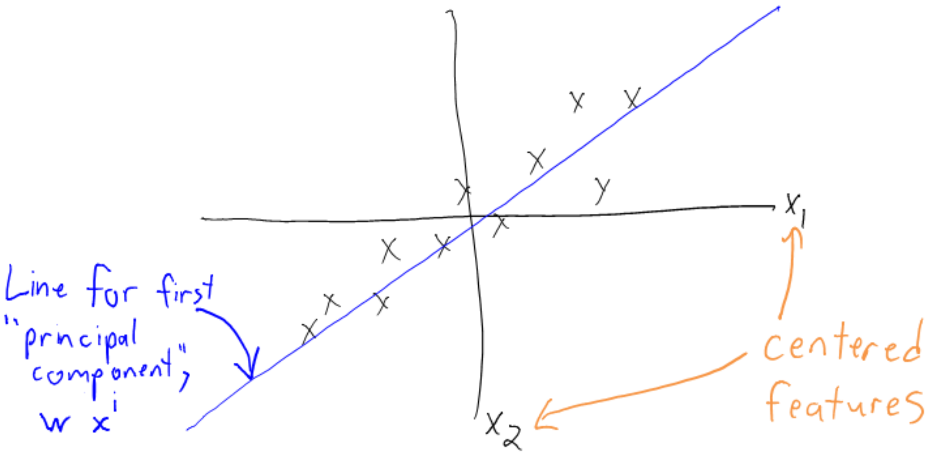
PCA in One Dimension



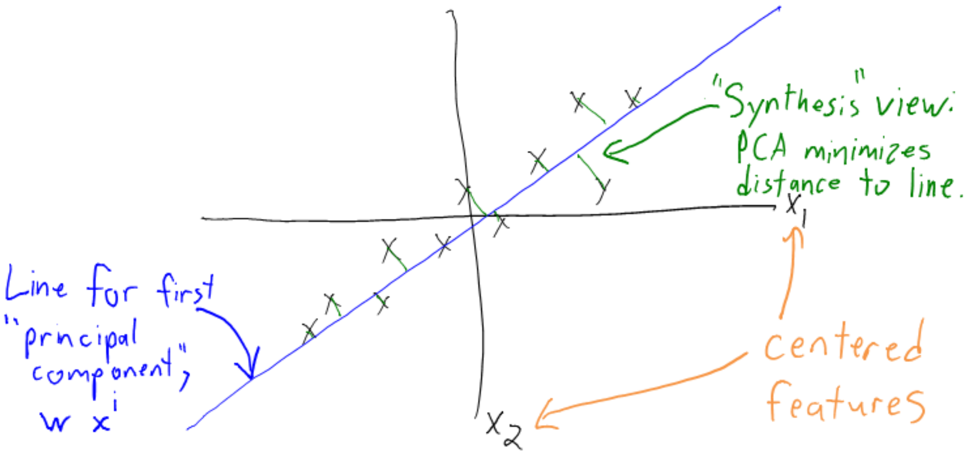
PCA in One Dimension



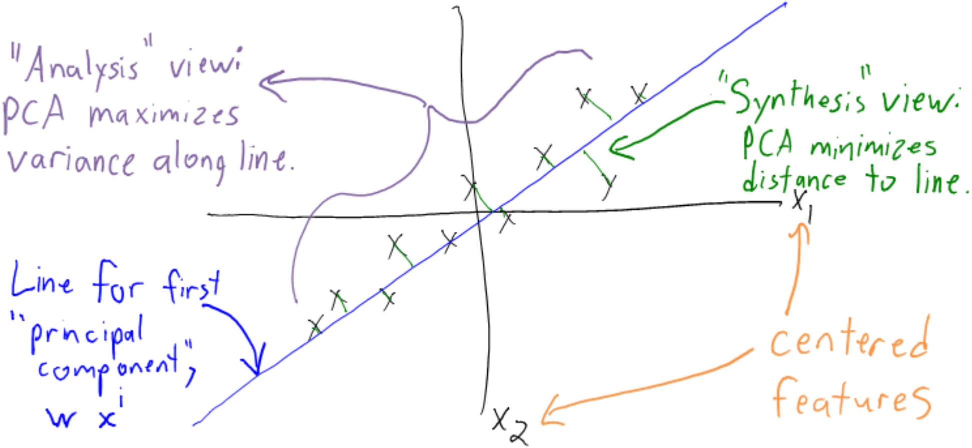
PCA in One Dimension



PCA in One Dimension



PCA in One Dimension



Probabilistic PCA

- With zero-mean ("centered") data, in PCA we assume that

$$x \approx W^T z.$$

- In probabilistic PCA we assume that

$$x \sim \mathcal{N}(W^T z, \sigma^2 I), \quad z \sim \mathcal{N}(0, I).$$

(we can actually use any Gaussian density for z)

Probabilistic PCA

- With zero-mean ("centered") data, in PCA we assume that

$$x \approx W^T z.$$

- In probabilistic PCA we assume that

$$x \sim \mathcal{N}(W^T z, \sigma^2 I), \quad z \sim \mathcal{N}(0, I).$$

(we can actually use any Gaussian density for z)

- Since z is hidden, our observed likelihood integrates over z,

$$p(x|W) = \int_z p(x, z|W) dz.$$

- Looks ugly, but can be computed due to the Gaussians assumptions:
  - This marginal distribution is Gaussian.

Manipulating Gaussians

- From the assumptions of the previous slide we have

$$p(x|z, W) \propto \exp\left(-\frac{(x - W^T z)^T (x - W^T z)}{2\sigma^2}\right), \quad p(z) \propto \exp\left(-\frac{z^T z}{2}\right).$$

Manipulating Gaussians

- From the assumptions of the previous slide we have

$$p(x|z,W) \propto \exp\left(-\frac{(x-W^Tz)^T(x-W^Tz)}{2\sigma^2}\right), \quad p(z) \propto \exp\left(-\frac{z^Tz}{2}\right).$$

- Multiplying and expanding we get

$$p(x,z|W) = p(x|z,W)p(z|W)$$

Manipulating Gaussians

- From the assumptions of the previous slide we have

$$p(x|z,W) \propto \exp\left(-\frac{(x-W^Tz)^T(x-W^Tz)}{2\sigma^2}\right), \quad p(z) \propto \exp\left(-\frac{z^Tz}{2}\right).$$

- Multiplying and expanding we get

$$\begin{aligned} p(x,z|W) &= p(x|z,W)p(z|W) \\ &= p(x|z,W)p(z) && (z \perp W) \end{aligned}$$

Manipulating Gaussians

- From the assumptions of the previous slide we have

$$p(x|z,W) \propto \exp\left(-\frac{(x-W^Tz)^T(x-W^Tz)}{2\sigma^2}\right), \quad p(z) \propto \exp\left(-\frac{z^Tz}{2}\right).$$

- Multiplying and expanding we get

$$\begin{aligned} p(x,z|W) &= p(x|z,W)p(z|W) \\ &= p(x|z,W)p(z) && (z \perp W) \\ &\propto \exp\left(-\frac{(x-W^Tz)^T(x-W^Tz)}{2\sigma^2} - \frac{z^Tz}{2}\right) \end{aligned}$$

Manipulating Gaussians

- From the assumptions of the previous slide we have

$$p(x|z,W) \propto \exp\left(-\frac{(x-W^Tz)^T(x-W^Tz)}{2\sigma^2}\right), \quad p(z) \propto \exp\left(-\frac{z^Tz}{2}\right).$$

- Multiplying and expanding we get

$$\begin{aligned}
p(x,z|W) &= p(x|z,W)p(z|W) \\
&= p(x|z,W)p(z) && (z \perp W) \\
&\propto \exp\left(-\frac{(x-W^Tz)^T(x-W^Tz)}{2\sigma^2} - \frac{z^Tz}{2}\right) \\
&= \exp\left(-\frac{x^Tx - x^TW^Tz - z^TWx + z^TWW^Tz}{2\sigma^2} + \frac{z^Tz}{2}\right)
\end{aligned}$$

Manipulating Gaussians

- So the "complete" likelihood satsifies

$$p(x, z|W) \propto \exp\left(-\frac{x^T x - x^T W^T z - z^T W x + z^T W W^T z}{2\sigma^2} + \frac{z^T z}{2}\right)$$

Manipulating Gaussians

- So the "complete" likelihood satsifies

$$
\begin{aligned}
p(x,z|W) &\propto \exp\left(-\frac{x^Tx - x^TW^Tz - z^TWx + z^TWW^Tz}{2\sigma^2} + \frac{z^Tz}{2}\right) \\
&= \exp\left(-\frac{1}{2}\left(x^T\left(\frac{1}{\sigma^2}I\right)x + x^T\left(\frac{1}{\sigma^2}W^T\right)z + z^T\left(\frac{1}{\sigma^2}W\right)x + z^T\left(\frac{1}{\sigma^2}WW^T + I\right)z\right)\right),
\end{aligned}
$$

Manipulating Gaussians

- So the "complete" likelihood satsifies

$$p(x,z|W) \propto \exp\left(-\frac{x^Tx - x^TW^Tz - z^TWx + z^TWW^Tz}{2\sigma^2} + \frac{z^Tz}{2}\right)$$
$$= \exp\left(-\frac{1}{2}\left(x^T\left(\frac{1}{\sigma^2}I\right)x + x^T\left(\frac{1}{\sigma^2}W^T\right)z + z^T\left(\frac{1}{\sigma^2}W\right)x + z^T\left(\frac{1}{\sigma^2}WW^T + I\right)z\right)\right),$$

- We can re-write the exponent as a quadratic form,

$$p(x,z|W) \propto \exp\left(-\frac{1}{2}\begin{bmatrix} z^T & x^T \end{bmatrix}\begin{bmatrix} \frac{1}{\sigma^2}WW^T + I & -\frac{1}{\sigma^2}W \\ -\frac{1}{\sigma^2}W^T & \frac{1}{\sigma^2}I \end{bmatrix}\begin{bmatrix} z \\ x \end{bmatrix}\right),$$

Manipulating Gaussians

- So the "complete" likelihood satsifies

$$p(x,z|W) \propto \exp\left(-\frac{x^Tx - x^TW^Tz - z^TWx + z^TWW^Tz}{2\sigma^2} + \frac{z^Tz}{2}\right)$$
$$= \exp\left(-\frac{1}{2}\left(x^T\left(\frac{1}{\sigma^2}I\right)x + x^T\left(\frac{1}{\sigma^2}W^T\right)z + z^T\left(\frac{1}{\sigma^2}W\right)x + z^T\left(\frac{1}{\sigma^2}WW^T + I\right)z\right)\right),$$

- We can re-write the exponent as a quadratic form,

$$p(x,z|W) \propto \exp\left(-\frac{1}{2}\begin{bmatrix} z^T & x^T \end{bmatrix}\begin{bmatrix} \frac{1}{\sigma^2}WW^T + I & -\frac{1}{\sigma^2}W \\ -\frac{1}{\sigma^2}W^T & \frac{1}{\sigma^2}I \end{bmatrix}\begin{bmatrix} z \\ x \end{bmatrix}\right),$$

- This has the form of a Gaussian distribution,

$$p(v|W) \propto \exp\left(-\frac{1}{2}v^T\Sigma^{-1}v\right),$$

with $v = \begin{bmatrix} z \\ x \end{bmatrix}$, $\mu = 0$, and $\Sigma^{-1} = \begin{bmatrix} \frac{1}{\sigma^2}WW^T + I & -\frac{1}{\sigma^2}W \\ -\frac{1}{\sigma^2}W^T & \frac{1}{\sigma^2}I \end{bmatrix}$.

Manipulating Gaussians

- Remember that if we write multivariate Gaussian in partitioned form,

$$\begin{bmatrix} x \\ z \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \mu_x \\ \mu_z \end{bmatrix}, \begin{bmatrix} \Sigma_{xx} & \Sigma_{xz} \\ \Sigma_{zx} & \Sigma_{zz} \end{bmatrix}\right),$$

then the marginal distribution $p(x)$ (integrating over z) is given by

$$x \sim \mathcal{N}(\mu_x, \Sigma_{xx}).$$

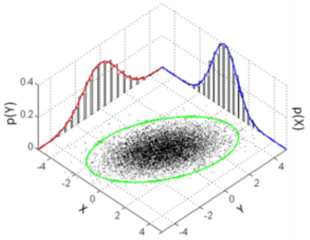
Manipulating Gaussians

- Remember that if we write multivariate Gaussian in partitioned form,

$$\begin{bmatrix} x \\ z \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \mu_x \\ \mu_z \end{bmatrix}, \begin{bmatrix} \Sigma_{xx} & \Sigma_{xz} \\ \Sigma_{zx} & \Sigma_{zz} \end{bmatrix}\right),$$

then the marginal distribution $p(x)$ (integrating over z) is given by

$$x \sim \mathcal{N}(\mu_x, \Sigma_{xx}).$$



https://en.wikipedia.org/wiki/Multivariate_normal_distribution

Manipulating Gaussians

- Remember that if we write multivariate Gaussian in partitioned form,

$$\begin{bmatrix} x \\ z \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \mu_x \\ \mu_z \end{bmatrix}, \begin{bmatrix} \Sigma_{xx} & \Sigma_{xz} \\ \Sigma_{zx} & \Sigma_{zz} \end{bmatrix}\right),$$

then the marginal distribution $p(x)$ (integrating over z) is given by

$$x \sim \mathcal{N}(\mu_x, \Sigma_{xx}).$$

- For probabilistic PCA we assume $\mu_x = 0$, but we partitioned Σ^{-1} instead of Σ.

Manipulating Gaussians

- Remember that if we write multivariate Gaussian in partitioned form,

$$\begin{bmatrix} x \\ z \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \mu_x \\ \mu_z \end{bmatrix}, \begin{bmatrix} \Sigma_{xx} & \Sigma_{xz} \\ \Sigma_{zx} & \Sigma_{zz} \end{bmatrix}\right),$$

then the marginal distribution $p(x)$ (integrating over z) is given by

$$x \sim \mathcal{N}(\mu_x, \Sigma_{xx}).$$

- For probabilistic PCA we assume $\mu_x = 0$, but we partitioned Σ^{-1} instead of Σ.
- To get Σ we can use a matrix inversion lemma,

$$\Sigma = \begin{bmatrix} \frac{1}{\sigma^2} W W^T + I & -\frac{1}{\sigma^2} W \\ -\frac{1}{\sigma^2} W^T & \frac{1}{\sigma^2} I \end{bmatrix}^{-1} = \begin{bmatrix} W^T W + \sigma^2 I & W^T \\ W & I \end{bmatrix},$$

which gives that solution to integrating over z is

$$x | W \sim \mathcal{N}(0, W^T W + \sigma^2 I).$$

Notes on Probabilistic PCA

- Negative log-likelihood of observed data has the form

$$-\log p(x|W) = \frac{n}{2}\mathrm{Tr}(SC) + \frac{n}{2}\log|C| + \text{const.},$$

where $C = W^T W + \sigma^2 I$ and $S = X^T X$.

Notes on Probabilistic PCA

- Negative log-likelihood of observed data has the form

$$-\log p(x|W) = \frac{n}{2}\text{Tr}(SC) + \frac{n}{2}\log|C| + \text{const.},$$

  where $C = W^TW + \sigma^2 I$ and $S = X^TX$.
- Not convex, but non-global stationary points are saddle points.
- Regular PCA is obtained as limit of $\sigma \to 0$ (as $\log|C| \to 0$).

Notes on Probabilistic PCA

- Negative log-likelihood of observed data has the form
$$-\log p(x|W) = \frac{n}{2}\text{Tr}(SC) + \frac{n}{2}\log|C| + \text{const.},$$
where $C = W^TW + \sigma^2 I$ and $S = X^TX$.
- Not convex, but non-global stationary points are saddle points.
- Regular PCA is obtained as limit of $\sigma \to 0$ (as $\log|C| \to 0$).
- Can reduce cost from $O(d^3)$ to $O(k^3)$ with matrix inversion/determinant lemmas:
  - Allows us to work with WW^T instead of W^TW.
- We can get $p(z|x, W)$ using that conditional of Gaussians is Gaussian.

Notes on Probabilistic PCA

- Negative log-likelihood of observed data has the form

$$-\log p(x|W) = \frac{n}{2}\text{Tr}(SC) + \frac{n}{2}\log|C| + \text{const.},$$

  where $C = W^TW + \sigma^2 I$ and $S = X^TX$.
- Not convex, but non-global stationary points are saddle points.
- Regular PCA is obtained as limit of $\sigma \to 0$ (as $\log|C| \to 0$).
- Can reduce cost from $O(d^3)$ to $O(k^3)$ with matrix inversion/determinant lemmas:
  - Allows us to work with WW^T instead of W^TW.
- We can get $p(z|x, W)$ using that conditional of Gaussians is Gaussian.

- We could consider different distribution for $x^i|z^i$ (but integrals are ugly):
  - E.g., Laplace of student if you want it to be robust.
  - E.g., logistic or softmax if you have discrete x^i_j.

Generalizations of Probabilistic PCA

- Why do we need a probabilistic interpretation of PCA?
  - Good excuse to play with Gaussian identities and matrix formulas?

Generalizations of Probabilistic PCA

- Why do we need a probabilistic interpretation of PCA?
  - Good excuse to play with Gaussian identities and matrix formulas?
- We now understand that PCA fits a Gaussian with restricted covariance:
  - Hope is that $W^TW + \sigma I$ is a good approximation of full covariance X^TX.

Generalizations of Probabilistic PCA

- Why do we need a probabilistic interpretation of PCA?
  - Good excuse to play with Gaussian identities and matrix formulas?
- We now understand that PCA fits a Gaussian with restricted covariance:
  - Hope is that $W^TW + \sigma I$ is a good approximation of full covariance X^TX.
  - We can do fancy things like mixtures of PCA models.

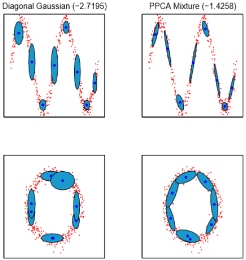


Figure 8: Comparison of an 8-component diagonal variance Gaussian mixture model with a mixture of PPCA model. The upper two plots give a view perpendicular to the major

http://www.miketipping.com/papers/met-mppca.pdf

- Lets us understand connection between PCA and factor analysis.

Factor Analysis

- Factor analysis (FA) is a method for discovering latent-factors.
- Historical applications are measures of intelligence and personality traits.
  - Some controversy, like trying to find factors of intelligence due to race.

(without normalizing for socioeconomic factors)

Trait	Description
Openness	Being curious, original, intellectual, creative, and open to new ideas.
Conscientiousness	Being organized, systematic, punctual, achievement-oriented, and dependable.
Extraversion	Being outgoing, talkative, sociable, and enjoying social situations.
Agreeableness	Being affable, tolerant, sensitive, trusting, kind, and warm.
Neuroticism	Being anxious, irritable, temperamental, and moody.

https://new.edu/resources/big-5-personality-traits

- But a standard tool and widely-used across science and engineering.

Factor Analysis

- FA approximates the original matrix by latent-variables Z and latent-factors W,

$$X \approx ZW.$$

- Which should sound familiar...

Factor Analysis

- FA approximates the original matrix by latent-variables Z and latent-factors W,

$$X \approx ZW.$$

- Which should sound familiar...
- Are PCA and FA the same?
  - Both are more than 100 years old.
  - People are still fighting about whether they are the same:
    - Doesn't help that some software packages run PCA when you call FA.

Google

pca vs. factor analysis

All Images Videos News Maps More ▾ Search tools

About 358,000 results (0.17 seconds)

[PDF] Principal Component Analysis versus Exploratory Factor ...
www2.sas.com/proceedings/sugi30/203-30.pdf ▾
by DD Suhr - Cited by 118 - Related articles
1. Paper 203-30. **Principal Component Analysis vs.** Exploratory **Factor Analysis**. Diana D. Suhr, Ph.D. University of Northern Colorado. Abstract. **Principal** ...

pca - What are the differences between Factor Analysis and ...
stats.stackexchange.com/.../what-are-the-differences-between-**factor-anal**... ▾
Aug 12, 2010 - **Principal Component Analysis** (PCA) and Common **Factor Analysis** (CFA) ..... differently one has to interpret the strength of loadings in PCA vs.

What are the differences between principal components ...
support.minitab.com/...**factor-analysis**/differences-between-**pca**-and-**facto**... ▾
Principal Components Analysis and **Factor Analysis** are similar because both procedures are used to simplify the structure of a set of variables. However, the ...

[PDF] Principal Components Analysis - UNT
https://www.unt.edu/rss/class/.../Principal%20Components%20**Analysis**.p... ▾
PCA vs. Factor Analysis. • It is easy to make the mistake in assuming that these are the same techniques, though in some ways exploratory factor analysis and ...

Factor analysis versus Principal Components Analysis (PCA)
psych.wisc.edu/henriques/**pca**.html ▾
Jun 19, 2010 - **Factor analysis versus PCA**. These techniques are typically used to analyze groups of correlated variables representing one or more common ...

[PDF] Principal Component Analysis and Factor Analysis
www.stats.ox.ac.uk/~ripley/MultAnal_HT2007/PC-FA.pdf ▾
where D is diagonal with non-negative and decreasing values and U and V ..... **Factor analysis** and PCA are often confused, and indeed SPSS has PCA as.

How can I decide between using principal components ...
https://www.researchgate.net/.../How_can_I_decide_between_using_prin... ▾
Factor analysis (FA) is a group of statistical methods used to understand and simplify patterns ... Retrieved from http://pareonline.net/getvn.asp?v=10&n=7 ... **Principal component analysis** (PCA) is a method of factor extraction (the second step ...

[PDF] Exploratory Factor Analysis and Principal Component An...
www.lesahoffman.com/948/948_Lecture2_EFA_**PCA**.pdf ▾
2 very different schools of thought on exploratory **factor analysis** (EFA) vs. principal components **analysis** (PCA): ➢ EFA and PCA are TWO ENTIRELY ...

Factor analysis - Wikipedia, the free encyclopedia
https://en.wikipedia.org/wiki/**Factor_analysis** ▾
Jump to Exploratory **factor analysis versus** principal components ... - [edit]. See also: **Principal component analysis** and Exploratory **factor analysis**.

[PDF] The Truth about PCA and Factor Analysis
www.stat.cmu.edu/~cshalizi/350/lectures/13/lecture-13.pdf ▾
Sep 28, 2009 - nents and **factor analysis**, we'll wrap up by looking at their uses and

PCA vs. Factor Analysis

- In probabilistic PCA we assume

$$x|z \sim \mathcal{N}(W^T z, \sigma^2 I), \quad z \sim \mathcal{N}(0, I),$$

and we obtain PCA as $\sigma \to 0$.

PCA vs. Factor Analysis

- In probabilistic PCA we assume

$$x|z \sim \mathcal{N}(W^T z, \sigma^2 I), \quad z \sim \mathcal{N}(0, I),$$

  and we obtain PCA as $\sigma \to 0$.
- In FA we assume

$$x|z \sim \mathcal{N}(W^T z, D), \quad z \sim \mathcal{N}(0, I),$$

  where D is a diagonal matrix.

PCA vs. Factor Analysis

- In probabilistic PCA we assume

$$x|z \sim \mathcal{N}(W^T z, \sigma^2 I), \quad z \sim \mathcal{N}(0, I),$$

  and we obtain PCA as $\sigma \to 0$.

- In FA we assume

$$x|z \sim \mathcal{N}(W^T z, D), \quad z \sim \mathcal{N}(0, I),$$

  where D is a diagonal matrix.

- The difference is that you can have a noise variance for each dimension.
- Repeating the previous exercise we get that

$$x \sim \mathcal{N}(0, W^T W + D).$$

PCA vs. Factor Analysis

- We can write non-centered versions of both models:
  - Probabilistic PCA:
$$x|z \sim \mathcal{N}(W^T z + \mu, \sigma^2 I), \quad z \sim \mathcal{N}(0, I),$$
  - Factor analysis:
$$x|z \sim \mathcal{N}(W^T z + \mu, D), \quad z \sim \mathcal{N}(0, I),$$

    where D is a diagonal matrix.

PCA vs. Factor Analysis

- We can write non-centered versions of both models:
  - Probabilistic PCA:

$$x|z \sim \mathcal{N}(W^T z + \mu, \sigma^2 I), \quad z \sim \mathcal{N}(0, I),$$

  - Factor analysis:

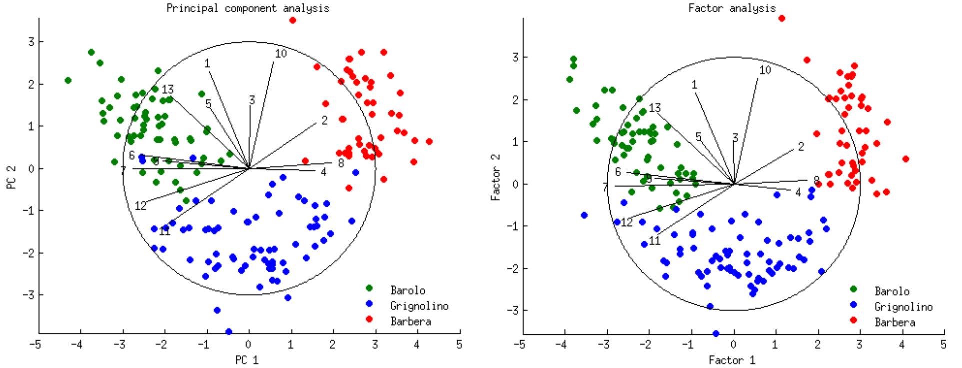
$$x|z \sim \mathcal{N}(W^T z + \mu, D), \quad z \sim \mathcal{N}(0, I),$$

  where D is a diagonal matrix.
- A different perspective is that these models assume

$$x = W^T z + \epsilon,$$

  where PPCA has $\epsilon \sim \mathcal{N}(\mu, \sigma^2 I)$ and FA has $\epsilon \sim \mathcal{N}(\mu, D)$.
- So in FA has extra degrees of freedom in variance of individual variables.

PCA vs. Factor Analysis



http://stats.stackexchange.com/questions/1576/what-are-the-differences-between-factor-analysis-and-principal-component-analysi

Remember in 340 that difference with PCA and ISOMAP/t-SNE was huge.

Factor Analysis Discussion

- No closed-form solution for FA, and can find different local optima.
- Unlike PCA, FA doesn't change if you scale variables.
  - FA doesn't chase large-noise features that are uncorrelated with other features.

Factor Analysis Discussion

- No closed-form solution for FA, and can find different local optima.
- Unlike PCA, FA doesn't change if you scale variables.
  - FA doesn't chase large-noise features that are uncorrelated with other features.
- Unlike PCA, FA changes if you rotate data.
- Similar to PCA, objective only depends on W^TW so you can rotate/mirror the factors

$$W^TW = W^T \underbrace{Q^TQ}_{I} W = (WQ)^T(WQ),$$

  for an orthogonal matrix Q.
  - So you can't interpret multiple factors as being unique.

Summary

- Monotonicity of EM: EM is guaranteed not to decrease likelihood.

Summary

- Monotonicity of EM: EM is guaranteed not to decrease likelihood.
- Kernel density estimation: Non-parametric continuous density estimation method.

Summary

- Monotonicity of EM: EM is guaranteed not to decrease likelihood.
- Kernel density estimation: Non-parametric continuous density estimation method.
- PCA is a classic method for dimensionality reduction.

Summary

- Monotonicity of EM: EM is guaranteed not to decrease likelihood.
- Kernel density estimation: Non-parametric continuous density estimation method.
- PCA is a classic method for dimensionality reduction.
- Probabilistic PCA is a continuous latent-variable probabilistic generalization.

Summary

- Monotonicity of EM: EM is guaranteed not to decrease likelihood.
- Kernel density estimation: Non-parametric continuous density estimation method.
- PCA is a classic method for dimensionality reduction.
- Probabilistic PCA is a continuous latent-variable probabilistic generalization.
- Factor analysis extends probabilistic PCA with different noise in each dimension.

- Next time: the algorithm we didn't cover in 340 from the list of
  "The 10 Algorithms Machine Learning Engineers Need to Know".

Bonus Slide: Mixture of Experts

- Classic generative model for supervised learning uses

$$p(y^i|x^i) \propto p(x^i|y^i)p(y^i),$$

and typically $p(x^i|y^i)$ is assumed by Gaussian (LDA) or independent (naive Bayes).

Bonus Slide: Mixture of Experts

- Classic generative model for supervised learning uses

$$p(y^i|x^i) \propto p(x^i|y^i)p(y^i),$$

  and typically $p(x^i|y^i)$ is assumed by Gaussian (LDA) or independent (naive Bayes).
- But we could allow more flexibility by using a mixture model,

$$p(x^i|y^i) = \sum_{c=1}^{k} p(z^i = c|y^i)p(x^i|z^i = c, y^i).$$

Bonus Slide: Mixture of Experts

- Classic generative model for supervised learning uses

$$p(y^i|x^i) \propto p(x^i|y^i)p(y^i),$$

  and typically $p(x^i|y^i)$ is assumed by Gaussian (LDA) or independent (naive Bayes).
- But we could allow more flexibility by using a mixture model,

$$p(x^i|y^i) = \sum_{c=1}^{k} p(z^i = c|y^i)p(x^i|z^i = c, y^i).$$

- Instead of a generative model, we could also take a mixture of regression models,

$$p(y^i|x^i) = \sum_{c=1}^{k} p(z^i = c|x^i)p(y^i|z^i = c, x^i).$$

- Called a "mixture of experts" model:
  - Each regression model is an "expert" for certain values of x^i.