Mixtures of Gaussians and EM

CSC411: Machine Learning and Data Mining, Winter 2017

Michael Guerzhoy
Unsupervised Learning

- Suppose the data (i.e., x’s) belongs to different classes, but we don’t have the labels (i.e., we don’t have the y’s)
- Won’t to characterize the different x’s somehow (e.g., “\(x^{(i)}\) belongs to cluster B,” there are 3 different clusters of data)
- Or to compute features that could be useful for classification (e.g., (1, 0, 0) if the x belongs to Cluster A, (0, 1, 0) if the x belongs to Cluster B, (0, 0, 1) if the x belongs to Cluster C)
  - If we can figure out how to compute those features using a large unlabelled dataset, we could then use them to perform supervised learning on a small labelled dataset
  - Like using the AlexNet features to classify faces
A Generative View

To generate a datapoint:

• Pick Cluster A with probability $P_A$, Cluster B with probability $P_B$, ...

• If we picked Cluster cl, sample random coordinates from $N(\mu_{cl}, \Sigma_{cl})$
A Generative View

• If the data is well-described as several “clouds” of points, we can generate a datapoint that looks like it was sampled from the training set by picking a cloud and then picking a coordinate from the cloud.

• “Clouds” can be conveniently described as multivariate Gaussians
Multivariate Gaussian: a quick intro (1)

Consider $\begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix} \sim \begin{pmatrix} N(\mu_1, \sigma_1^2) \\ N(\mu_2, \sigma_2^2) \\ \vdots \\ N(\mu_n, \sigma_n^2) \end{pmatrix}$

• Here, we are sampling an n-dimensional point, with every dimension sampled independently.

• If we sample a lot of points, we’ll get something that looks like a cloud, where large $\sigma_k$ means that the cloud is “wider” along dimension k.

• The cloud will be “axis-aligned,” in the sense that it won’t be tilted.
Multivariate Gaussian: a quick intro (2)

- The cloud will not look like this:

- But it could look like this:
Multivariate Gaussian: a quick intro (3)

• The mathematical way to describe an “axis-aligned” cloud is to say
  • $\text{Cov}(x_i, x_j) = 0$ for $i \neq j$
  • I.e., the coordinates along axes $i$ and $j$ are uncorrelated

• A multivariate Gaussian distribution allows as to specify the covariances between coordinates along axes $i$ and $j$.
  • Reminder: $\text{Cov}(x_1, x_2) = E[(x_2 - \mu_1)(x_2 - \mu_2)]$
    $$\approx \frac{1}{N} \sum (x_1^{(i)} - \bar{x}_1)(x_2^{(i)} - \bar{x}_2)$$
Multivariate Gaussian: a quick intro (4)

• Specify the covariance matrix $\Sigma$:

$$\Sigma = \begin{pmatrix}
\text{Cov}(x_1, x_1) & \cdots & \text{Cov}(x_1, x_n) \\
\vdots & \ddots & \vdots \\
\text{Cov}(x_n, x_1) & \cdots & \text{Cov}(x_n, x_n)
\end{pmatrix}$$

• We can have a multivariate Gaussian distribution that’s specified by

$$X \sim N(\mu, \Sigma)$$

• It generates a cloud of points, but this time the coordinates might be correlated
Multivariate Gaussian: a quick intro (5)

• Suppose $X \sim \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 1 & .2 \\ .2 & 1 \end{pmatrix}$

• That means that
  • $Var(x_1) = Var(x_2) = Cov(x_1, x_1) = Cov(x_2, x_2) = 1$
  • $Cov(x_1, x_2) = .2$

• The larger $x_1$, the larger we expect $x_2$ to be
Multivariate Gaussian: a quick intro (6)

• The density of the multivariate Gaussian:

\[
f(x; \mu, \Sigma) = \frac{1}{\sqrt{(2\pi)^k |\Sigma|}} \exp\left(-\frac{1}{2} (x - \mu)^T \Sigma^{-1} (x - \mu) \right)
\]

• \(k\) is the dimensionality of \(x\) (so \(\dim(\Sigma) = k \times k\))
• \(|\Sigma| = \det(\Sigma)\)
Learning One Gaussian

• We observe a bunch of points \( D = \{x^{(1)}, x^{(2)}, \ldots \} \)

• We assume that they were all generated by a single (multivariate) Gaussian

• We can learn it using maximum likelihood: maximize the probability \( P(D|\theta) \) that the data was generated using a Gaussian parameterized by \( \theta = \{\mu, \Sigma\} \).

• We can show (using calculus) that the ML estimates are:

\[
\hat{\mu} = \frac{1}{m} \sum_{i=1}^{m} x^{(i)}, \quad \hat{\Sigma} = \frac{1}{m} \sum_{i=1}^{m} (x^{(i)} - \bar{x})(x^{(i)} - \bar{x})^T
\]
\[ \hat{\mu} = \frac{1}{m} \sum_{i=1}^{m} x^{(i)}, \hat{\Sigma} = \frac{1}{m} \sum_{i=1}^{m} (x^{(i)} - \bar{x})(x^{(i)} - \bar{x})^T \]

• \( \hat{\mu} = \frac{1}{m} \sum_{i=1}^{m} x^{(i)} \) makes sense: the mean of the Gaussian is the mean of the vectors \( x^{(i)} \)

• The \((k, n)\)-th component of \( \frac{1}{m} \sum_{i=1}^{m} (x^{(i)} - \bar{x})(x^{(i)} - \bar{x})^T \) the estimated Cov\((x_k, x_n), \frac{1}{m} \sum_{i=1}^{m} (x_k^{(i)} - \bar{x}_k)(x_n^{(i)} - \bar{x}_n)\)
Mixture of Gaussians

- \( P(x|\pi, \mu, \Sigma) = \sum_{cl} P(x|\mu, \Sigma, cl)P(cl|\pi) \) by the law of total probability
  - Weighted sum of the likelihoods for all the clusters, weighted by the probabilities of the clusters

- \( P(x|\pi, \mu, \Sigma) = \sum_{cl} P(x|\mu, \Sigma, cl)P(cl|\pi) = \sum_{cl} P(x|\mu_{cl}, \Sigma_{cl})\pi_{cl} \)
Learning a Mixture of Gaussians

• Let $z^{(i)}$ by the cluster to which point $i$ is assigned

• If we knew all the $z^{(i)}$, we could learn the Gaussians one-by-one. But we don’t. Instead, we can try to estimate

$$w_{cl}^{(i)} = p(z^{(i)} = cl|x^{(i)}, \pi, \mu, \Sigma) = \frac{P(x^{(i)}|\mu_{cl}, \Sigma_{cl})\pi_{cl}}{P(x^{(i)}|\pi, \mu, \Sigma)} \propto P(x^{(i)}|\mu_{cl}, \Sigma_{cl})\pi_{cl}$$

• But we don’t know $\mu$, $\Sigma$, $\pi$ either! But if we estimate the z’s, it’s easy to estimate $\mu$, $\Sigma$, $\pi$. 
Learning a Mixture of Gaussians

• E-step:
• Want to estimate the cluster assignments $z^{(i)}$.

\[
\phi_{cl}^{(i)} = P(x^{(i)}|\mu_{cl}, \Sigma_{cl})\pi_{cl}
\]

\[
w_{cl}^{(i)} = p(z^{(i)} = cl|x^{(i)}, \pi, \mu, \Sigma) = \frac{P(x^{(i)}|\mu_{cl}, \Sigma_{cl})\pi_{cl}}{P(x^{(i)}|\pi, \mu, \Sigma)} \propto \phi_{cl}^{(i)}
\]

\[
w_{cl}^{(i)} = \frac{\phi_{cl}^{(i)}}{\sum_{c'l, \phi_{cl}^{(i)}}}
\]
Learning a Mixture of Gaussians

• M-step: Assume probabilistic cluster assignments were done

\[
\pi_{cl} = \frac{1}{m} \sum_i w^{(i)}_{cl}
\]

\[
\mu_{cl} = \frac{\sum_i w^{(i)}_{cl} x^{(i)}}{\sum_i w^{(i)}_{cl}}
\]

\[
\Sigma_{cl} = \frac{\sum_i w^{(i)}_{j} (x^{(i)} - \mu_{cl}) (x^{(i)} - \mu_{cl})^T}{\sum_i w^{(i)}_{cl}}
\]
Learning a Mixture of Gaussians

• Start with an initial guess of $\pi$, $\mu$, $\Sigma$

• Repeat:
  • Perform E-step to estimate the (probabilistic) cluster assignments of each point
    
    \[ w_{cl}^{(i)} = p(z^{(i)} = cl|x^{(i)}, \pi, \mu, \Sigma) \]

  • Assume cluster assignments, and re-estimate $\pi$, $\mu$, $\Sigma$ based on them
Learning a Mixture of Gaussians

• Very easy to get stuck in local optima

• Example:
  • A Gaussian whose variance is very small, and whose mean is very close to one point $x$
  • The E-step will only assign $x$ to that Gaussian since the variance of the Gaussian is very small so the likelihood for any other point is small
  • The M-step will make the mean exactly equal to $x$, and make the variance even smaller

• Solution: start with Gaussians with large variances
Learning a Mixture of Gaussians

• How do we select the number of clusters?
• Try different numbers of clusters, select the number of clusters that maximizes the probability density of the validation set
  • Imagine fitting a very small-variance Gaussian to every point in the training set: this would give a very small probability density to the validation set
K-means

• K means is an algorithm for finding centres of clusters
• Simpler than Mixture of Gaussians, but the same idea
K-means

- Assignment step: assign each datapoint to the closest cluster
- Refitting step: Move each cluster center to the average of the points assigned to the cluster

Slide from Geoff Hinton
Why K-means converges

• Whenever an assignment is changed, the sum squared distances of datapoints from their assigned cluster centers is reduced.

• Whenever a cluster center is moved the sum squared distances of the datapoints from their currently assigned cluster centers is reduced.

• If the assignments do not change in the assignment step, we have converged.
K-means: local optima

- You could get back local optima with k-means
- Try multiple starting points
  - How to evaluate how good the result is?
Speeding up Learning: MoG

• Run k-means first, initialize the means of the Gaussians to be the means obtained using k-means