Expectation Maximization

DORU ARFIRE

Papers We Love, April 11th 2016

The Plan

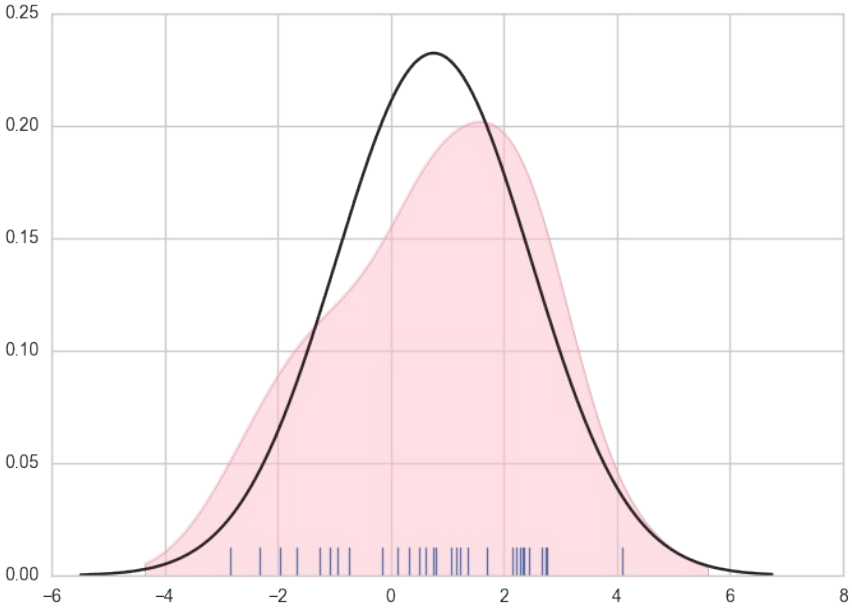
- Gaussian Mixture Models example
- History
- Formal definition
- Usage in practice
- Applications
- GMM demonstration
- Q&A

A simple problem

- We draw $X_1, X_2, \ldots, X_n$ samples from $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\sigma})$
- We need to estimate $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$
- Simple

$$\mu = \frac{1}{N} \sum_{k=1}^{N} X_k$$

$$\sigma^2 = \frac{1}{N} \sum_{k=1}^{N} (X_k - \mu)^2$$

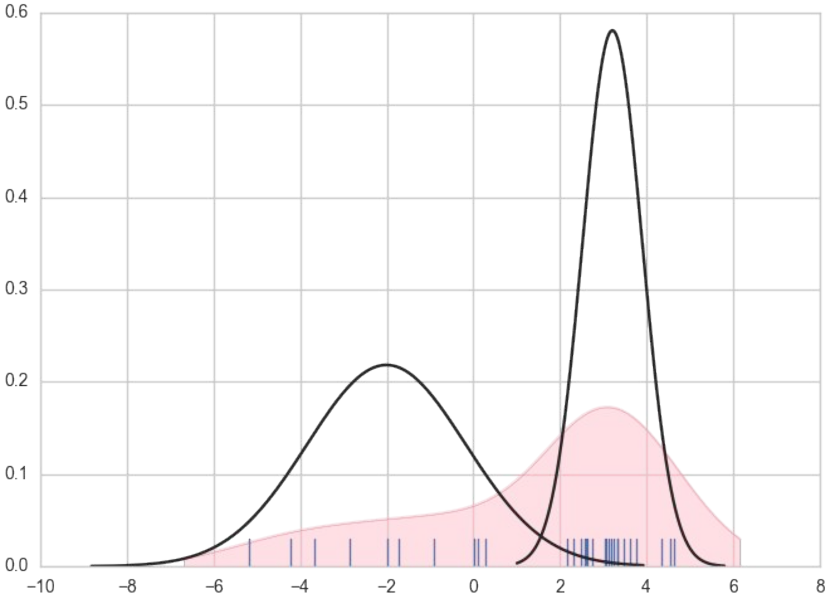


Maximum Likelihood Estimation

- Given X_1, X_2, … from a distribution $\mathcal{D}(\boldsymbol{\theta})$
- The Likelihood Function: $\mathscr{L}(\boldsymbol{\theta} \mid \mathbf{X}) = P(\mathbf{X} \mid \boldsymbol{\theta}) = \prod P(X_k \mid \boldsymbol{\theta}) = \prod P_{\boldsymbol{\theta}}(X_k)$
  - We usually work with $\log\mathscr{L}(\boldsymbol{\theta} \mid \mathbf{X})$
  - For exponential family distributions $\log\mathscr{L}(\boldsymbol{\theta} \mid X)$ is strictly concave
- A function of the parameter $\boldsymbol{\theta}$, not X
- Maximum Likelihood Estimation
  - Find $\boldsymbol{\theta}$ that maximizes□$\mathscr{L}(\boldsymbol{\theta} \mid \mathbf{X})$
  - Depending on distribution, optimize using:
    - Shortcut estimator formulas
    - Derivatives, then solve equations for $\boldsymbol{\theta}$
    - Numeric optimization (Gradient Descent & friends)
    - Monte-Carlo methods

Gaussian Mixture Models

- We draw $X_1, X_2, \ldots, X_n$ from either $\mathcal{N}_1(\boldsymbol{\mu}_1, \boldsymbol{\sigma}_1)$ or $\mathcal{N}_2(\boldsymbol{\mu}_2, \boldsymbol{\sigma}_2)$
- We don't know
  - which distribution each point came from
  - the parameters $\boldsymbol{\theta} = (\boldsymbol{\mu}_1, \boldsymbol{\sigma}_1, \boldsymbol{\mu}_2, \boldsymbol{\sigma}_2)$
- We need to find $\boldsymbol{\theta}$
- Bonus Points: figure out what distribution each point comes from
- Basically clustering



Gaussian Mixture Models

- Model X_i belonging to each distribution using a ***latent*** random variable Y_i
  - $P(Y_i=j)$ is probability that X_i came from $\mathcal{N}_j$
- Likelihood becomes ***expected likelihood***
  - $E_Y[\mathcal{L}(\boldsymbol{\theta} \mid X)] = \prod (P(Y_k=1)\, P(X_k|\boldsymbol{\mu}_1, \boldsymbol{\sigma}_1) + P(Y_k=2)\, P(X_k|\boldsymbol{\mu}_2, \boldsymbol{\sigma}_2))$
  - $E_Y[\log\mathcal{L}(\boldsymbol{\theta} \mid X)] = \sum[P(Y_k=1) \log P(X_k|\boldsymbol{\mu}_1, \boldsymbol{\sigma}_1) + P(Y_k=2) \log P(X_k|\boldsymbol{\mu}_2, \boldsymbol{\sigma}_2)]$
- No pretty closed form, not concave anymore

Gaussian Mixture Models

- Model X_i belonging to each distribution using a ***latent*** random variable Y_i
  - $P(Y_i=j)$ is probability that X_i came from $\mathcal{N}_j$
- Likelihood becomes ***expected likelihood***
  - $E_Y[\mathcal{L}(\boldsymbol{\theta} \mid X)] = \prod (P(Y_k=1)\, P(X_k|\boldsymbol{\mu}_1, \boldsymbol{\sigma}_1) + P(Y_k=2)\, P(X_k|\boldsymbol{\mu}_2, \boldsymbol{\sigma}_2))$
  - $E_Y[\log\mathcal{L}(\boldsymbol{\theta} \mid X)] = \sum[P(Y_k=1) \log P(X_k|\boldsymbol{\mu}_1, \boldsymbol{\sigma}_1) + P(Y_k=2) \log P(X_k|\boldsymbol{\mu}_2, \boldsymbol{\sigma}_2)]$
- No pretty closed form, not concave anymore
- Couldn't we use Gradient Descent?

Gaussian Mixture Models

- Model X_i belonging to each distribution using a ***latent*** random variable Y_i
  - $P(Y_i=j)$ is probability that X_i came from $\mathcal{N}_j$
- Likelihood becomes ***expected likelihood***
  - $E_Y[\mathcal{L}(\boldsymbol{\theta} \mid X)] = \prod (P(Y_k=1)\, P(X_k|\boldsymbol{\mu}_1, \boldsymbol{\sigma}_1) + P(Y_k=2)\, P(X_k|\boldsymbol{\mu}_2, \boldsymbol{\sigma}_2))$
  - $E_Y[\log\mathcal{L}(\boldsymbol{\theta} \mid X)] = \sum[\color{red}{P(Y_k=1)}\, \log P(X_k|\color{red}{\boldsymbol{\mu}_1, \boldsymbol{\sigma}_1}) + \color{red}{P(Y_k=2)}\, \log P(X_k|\color{red}{\boldsymbol{\mu}_2, \boldsymbol{\sigma}_2})]$
- No pretty closed form, not concave anymore
- Couldn't we use Gradient Descent? Of course, but:
  - Many parameters: $O(|X| + |\boldsymbol{\theta}|)$
  - Slow to converge, in practice
- Can use a faster method, specific for likelihood functions

GMM: Expectation Maximization

- Chicken and Egg problem
  - If we know Y_1, Y_2, … it's easy to calculate $\boldsymbol{\theta}$ (do MLE)
  - If we know $\boldsymbol{\theta}$ it's easy to calculate $P(Y_k)$: $P(Y_k=1) \sim \mathcal{L}(\boldsymbol{\mu}_1, \sigma_1 \mid X_k) = P(X_k \mid \boldsymbol{\mu}_1, \sigma_1)$
- Expectation Maximization algorithm intuition
  - Start with a guess for $\boldsymbol{\theta}$
  - E-step: Using current $\boldsymbol{\theta}$ estimate, compute $P(Y_k)$
  - M-step: Using current $P(Y_k)$ estimate $\boldsymbol{\theta}$ (MLE)
  - Repeat E-step and M-step until convergence
- Guaranteed to converge to a stationary point
  - Most likely a local maximum

GMM: Expectation Maximization

- E-step:

$$P(Y_k = j) = \frac{P(Y_k = j \mid \mu_j, \sigma_j)}{\sum_i P(Y_k = i \mid \mu_i, \sigma_i)} = \frac{\frac{1}{\sigma_j \sqrt{2\pi}} exp(-\frac{(X_k - \mu_j)^2}{2\sigma_j^2})}{\sum_i \frac{1}{\sigma_i \sqrt{2\pi}} exp(-\frac{(X_k - \mu_i)^2}{2\sigma_i^2})}$$

- M-step: expected log likelihood

$$E_Y[log\mathscr{L}(\theta|X)] = -\frac{1}{2} \sum_{k=1}^{N} \sum_{j=1,2} P(Y_k=j)[ln\sigma_j^2 + ln2\pi + \frac{(X_k - \mu_j)^2}{\sigma_j}]$$

GMM: Expectation Maximization

- M-step

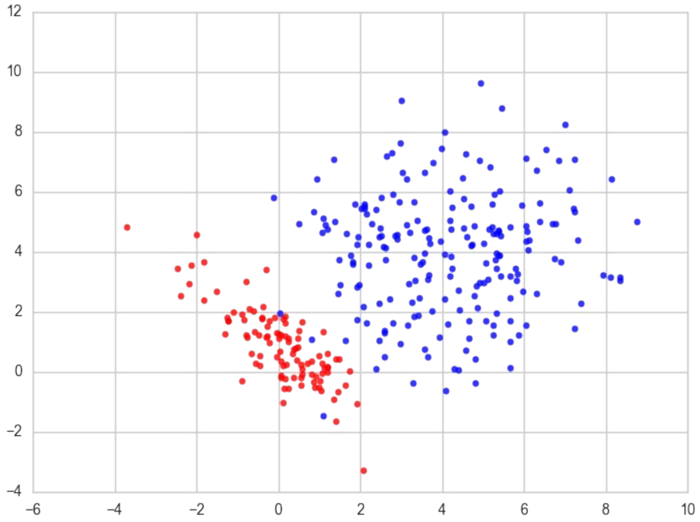
$$E_Y[log\mathscr{L}(\theta|X)] = -\frac{1}{2}\sum_{k=1}^{N}\sum_{j=1,2}P(Y_k=j)[ln\sigma_j{}^2+ln2\pi+\frac{(X_k-\mu_j)^2}{\sigma_j}]$$

- Taking derivatives and solving equations we get

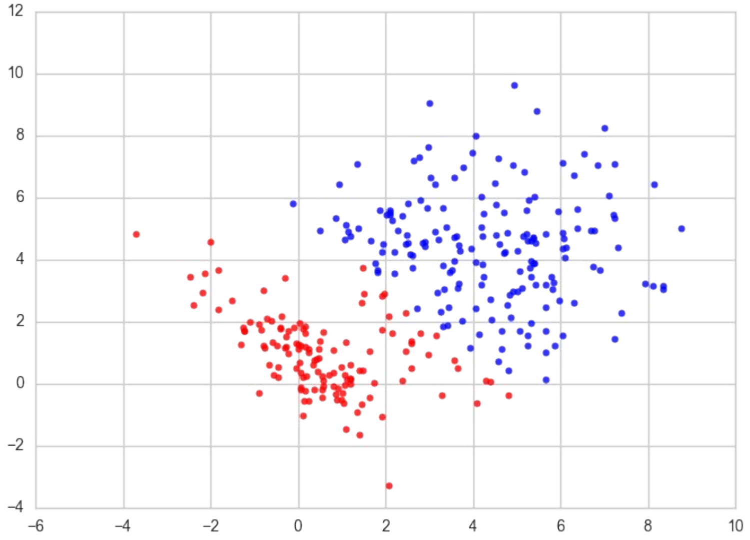
$$\mu_j = \frac{\sum_k P(Y_k=j)X_k}{\sum_k P(Y_k=j)} \qquad \sigma_j^2 = \frac{\sum_k P(Y_k=j)(X_k-\mu_j)^2}{\sum_k P(Y_k=j)}$$

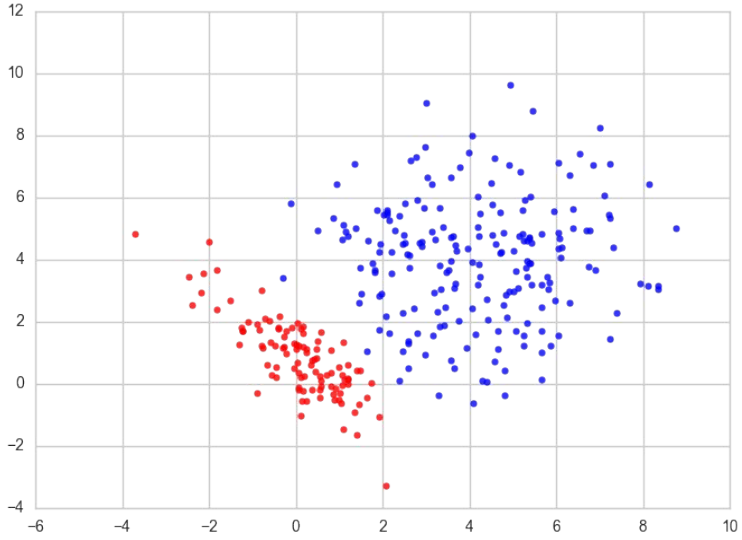
GMM: Expectation Maximization

- Similar to K-Means
    - 2-step iteration
- Uses a *soft* assignment to clusters
- Supports multiple parameters ($\boldsymbol{\mu}$ and $\boldsymbol{\theta}$)
- Can find clusters of different *radius*



GMM: EM vs K-Means





Expectation Maximization

- Invented independently by multiple authors
- Estimation of parameters when we have incomplete data
  - We never observe some variables
  - We sometimes don’t observe variables
- Hartley, H.O. - *Maximum Likelihood Estimation From Incomplete Data*(1958)
- Dempster, Laird, Rubin - *Maximum Likelihood Estimation From Incomplete Data Via the EM Algorithm*(1977)
  - Introduced the EM name
  - Incorrect convergence proof
- Wu, C. F. Jeff - On the Convergence Properties of the EM Algorithm (1983)
  - Proof of convergence

Expectation Maximization

- Given observed data **X**, unobserved data **Z** and parameters $\boldsymbol{\theta}$
- We need to find $\boldsymbol{\theta}$ and **Z** that maximize $\mathscr{L}(\theta;X)=P(X|\theta)=\sum_{Z} P(X,Z|\theta)$
- E-step
  - Keeps $\boldsymbol{\theta}$ constant
  - Computes the coefficients for the *expected value* for the log likelihood
  - $Q(\theta|\theta^t) = E_{Z|X,\theta^t}(log\mathscr{L}(\theta;X,Z))$
- M-step
  - Choose parameters that maximize Q: $\theta^{t+1} = \underset{\theta}{argmax}\ Q(\theta|\theta^t)$
- Actual implementation depends on the distributions involved

Expectation Maximization: Why it works?

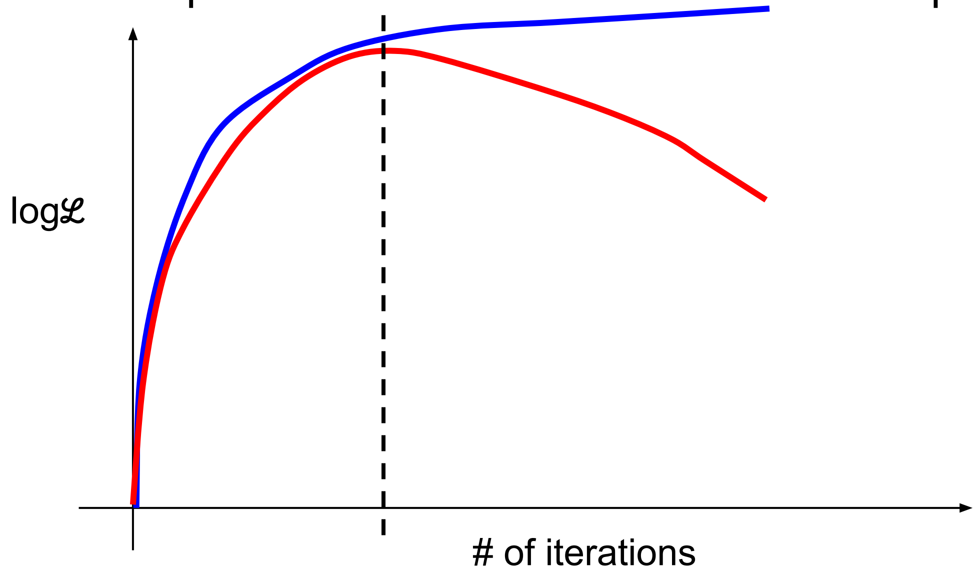
- Guarantees that
  - Each iteration improves the expected likelihood
  - $\mathcal{L}(\boldsymbol{\theta}^{t+1}; X) \geq \mathcal{L}(\boldsymbol{\theta}^{t}; X)$
  - If $\boldsymbol{\theta}^{t+1} = \boldsymbol{\theta}^{t}$ then we have reached a stationary point
- Faster convergence than Gradient Descent
  - Each step solves MLE problem => faster convergence
  - MLE optimizes a simpler function: $|\boldsymbol{\theta}|$ vs. $|X| + |\boldsymbol{\theta}|$
  - Specific algorithm for likelihood functions

EM in practice

- Not all local-maxima are the same
  - The more missing data we have the more local maxima we find
- Can be combined with simulated annealing for *probing* for global maxima
- Sensitive to initialization
  - Multiple restarts, choose best local maximum
  - Initialize using domain knowledge
  - Initialize using a simpler algorithm (K-Means)

EM in practice: Overfitting

- Running it to convergence may overfit data
  - Use a separate data set to decide when to stop



Applications: Missing Data Imputation

- When some samples miss values for certain features
  - Recording error
  - Non-replies in surveys
  - Too expensive to record
- Suboptimal methods of handling missing data
  - Dropping rows with missing data
  - Mean imputation

Applications: Missing Data Imputation

- Different patterns of missing data
  - Missing Completely At Random; includes unobserved variables
    - Income missing at random
  - Missing At Random
    - Prob of missing an observation depends on other observed variables
    - Income missing depending on age, but we know age
  - Missing Not At Random
    - Depends on both observed and unobserved variables
    - Income missing depending on the income level

Applications: Missing Data Imputation

- Applicable to MAR (MCAR $\subseteq$ MAR)
- E-step
  - We know current means and covariance matrix ($\boldsymbol{\theta}$)
  - Impute missing data using linear models
  - Unobserved variables as a linear combination of the observed variables (regression analysis)
- M-step
  - Recompute parameters based on complete data

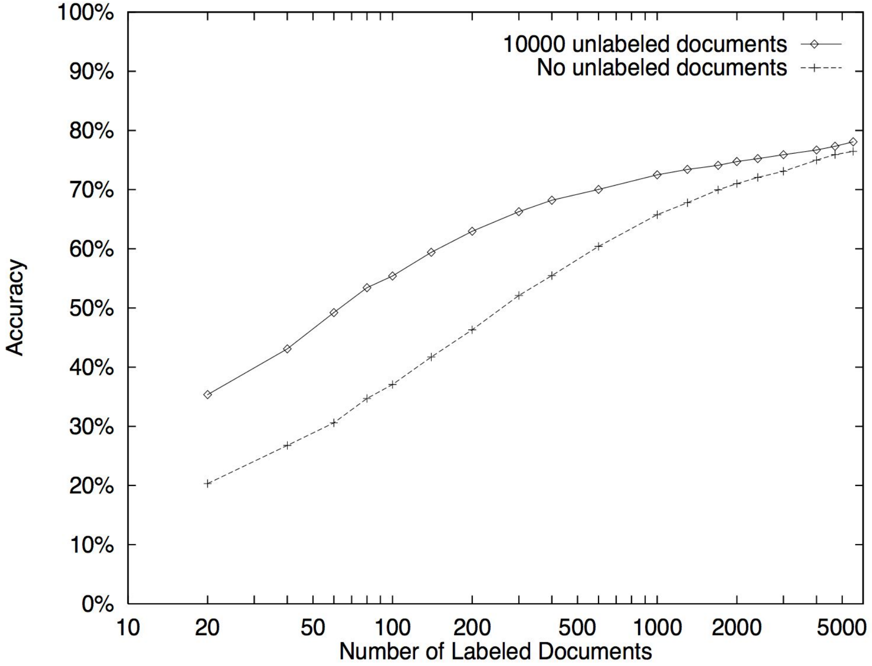
Applications: Bayesian Clustering

- Learning with latent variables
  - Variables that we never observe (a form of MCAR)
  - Can simplify our model
  - Often encode the most interesting information
- Bayesian clustering
  - Unsupervised learning of classes from data
  - Cluster examples based on shared feature values
  - News-site users segmentation based on viewed articles
  - Unsupervised or Semi-supervised Naïve Bayes

Applications: Semi-supervised Naïve Bayes

Kamal Nigam, Andrew McCallum and Tom Mitchell. Semi-supervised Text Classification Using EM

- We have both labeled and unlabeled data
- Train classifier on labeled data
- E-step:
  - Use classifier to find $P(C_i \mid X_j)$ for all classes and all unlabeled examples
- M-step:
  - Train classifier on both labeled and unlabelled data
- Repeat EM while likelihood increases
- Gives better performance than using only labeled data



Other applications

- Hidden Markov Models
  - Generative probabilistic graphical model: $P(Y, X) = P(Y \mid X)\, P(X)$
  - Used in speech recognition, natural language processing
  - Baum-Welch algorithm is a variation of EM
- Conditional Random Fields
  - Discriminative probabilistic graphical model: $P(Y \mid X)$
  - Sequential logistic regression
  - Used in spoken language understanding, POS tagging, NLP chunking
  - Trained using EM
- Many many other applications

References

- Further study
  - Chuong B.D., Batzoglou S. -- What is the expectation maximization algorithm? (the infamous coins example)
  - Daphne Koller -- “Probabilistic Graphical Models” Coursera course, week 21
- Implementations
  - Mostly for Gaussian Mixtures
  - Python -- sklearn.mixture.GMM
  - R -- mclust, EMCluster
  - Apache Spark -- org.apache.spark.mllib.clustering.GaussianMixture
  - Apache Mahout

Q&A