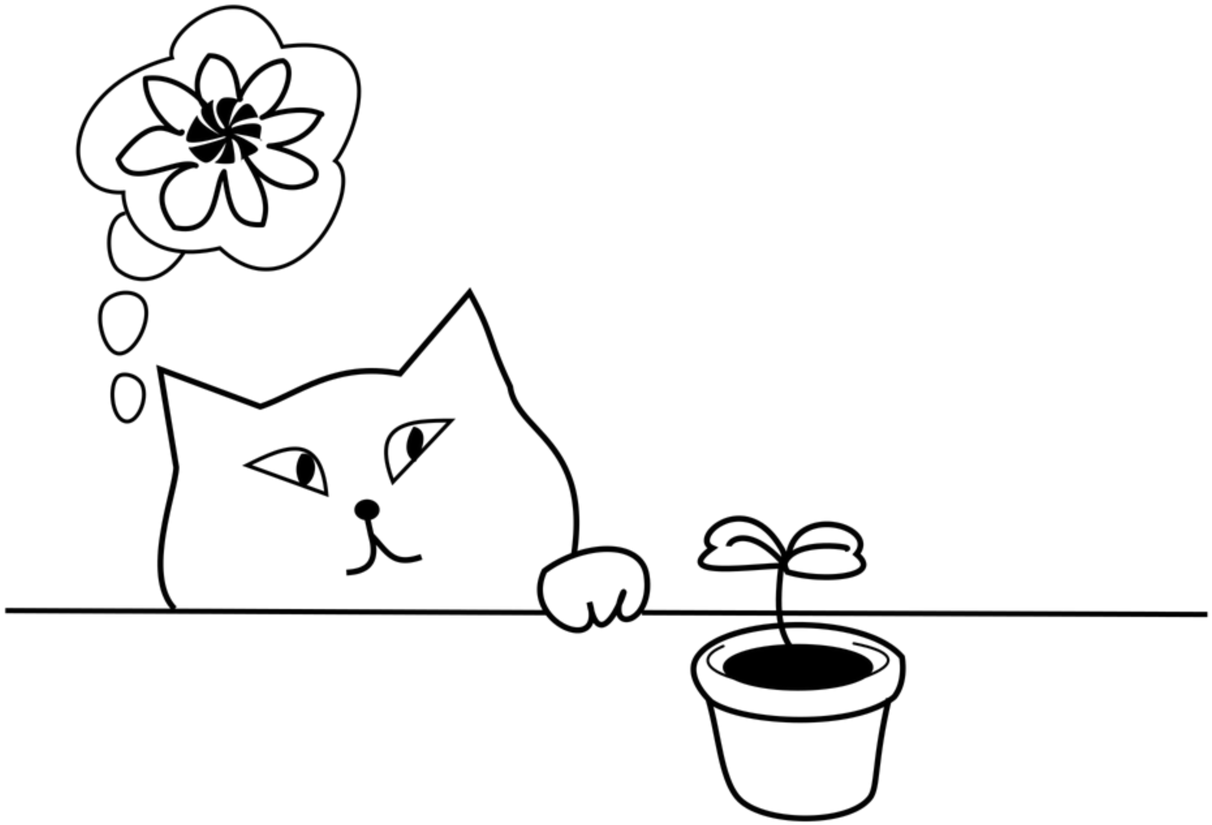
The Expectation-Maximization Algorithm:

Bernoulli Mixture Models Case Study and the General Case



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

Michael Guerzhoy and Lisa Zhang

Naïve Bayes: Review

- Training data:
  $$\left(x^{(1)}, y^{(1)}\right), \left(x^{(2)}, y^{(2)}\right), \ldots, \left(x^{(m)}, y^{(m)}\right)$$
  - x: an p-dimensional vector of binary variables
  - y: a discrete label
- Assumption:
  $$P\left(x_1, \ldots, x_p \middle| y = c\right) = \Pi_{i=1}^{p} \, p\left(x_j \middle| y = c\right)$$
- Estimate:
  $$P\left(x_j = 1 \middle| y = c\right) \approx \frac{count(x_j = 1, y = c)}{count(y = c)}$$
  $$P(y = c) \approx \frac{count(y = c)}{m}$$

Parameters:
$$\theta_{j,c} = P\left(x_j = 1 \middle| y = c\right)$$
$$\pi_c = P(y = c)$$

- Predict:
  $$P(y = c|x) = \frac{P(y=c)P(x|y=c)}{\sum_{c\prime} P(y=c')P(x|y=c\prime)}$$

Naïve Bayes: Num. of Parameters

- $\theta_{j,c} = P\left(x_j = 1 \middle| y = c\right)$
  - $\#classes \times \dim(x)$ parameters
  - $P\left(x_j = 0 \middle| y = c\right) = 1 - \theta_{ij\,c}$
- $\pi_c = P(y = c)$
  - $(\#classes - 1)$ parameters
  - $\pi_1 = 1 - \sum_{c'=2..\#classes} \pi_{c'}$
- A total of $\#classes \times \dim(x) + \#classes - 1$ parameters to estimate

What if we don't know the labels?

- If we know the parameters, we can guess the labels

$$P(y = c|x) = \frac{P(y=c)P(x|y=c)}{\sum_{c\prime} P(y=c')P(x|y=c\prime)}$$

- Can guess $y = 1$ if $P(y = c|x) > 0.5$, or just be happy with the probability that $y = c$: the expectation of an indicator variable that checks if the class is c
  - $E[I[y = c]|x] = P(y = c|x)$
- If we know the labels, we can estimate the parameters

$$I[y = c] = \begin{cases} 1, y = c \\ 0, otherwise \end{cases}$$

Expectation-Maximization

- Start with a random guess of the parameters θ and π
- Repeat:

E-step

  - For each example i in the training set, compute $E_{\theta,\pi}\left[I\left[y^{(i)}=c\right]|x^{(i)}\right] = P_{\theta,\pi}\left(y^{(i)}=c\middle|x^{(i)}\right)$ for every class c
  - Compute the expected number of examples for every class c and feature j

$$\widehat{count}\left(x_j=1, y=c\right) = E_{\theta,\pi}\left[\sum_{i|x_j^{(i)}=1} I\left[y^{(i)}=c\right]|x^{(i)}\right] = \sum_{i|x_j^{(i)}=1} E_{\theta,\pi}\left[I\left[y^{(i)}=c\right]|x^{(i)}\right]$$

$$\widehat{count}\left(y=c\right) = E_{\theta,\pi}\left[\sum_{i} I\left[y^{(i)}=c\right]|x^{(i)}\right]$$

M-step

  - Re-estimate the θ and π using the new counts

E-step

$$E_{\theta,\pi}\left[I\left[y^{(i)} = c\right]\middle|x^{(i)}\right] = P_{\theta,\pi}\left(y^{(i)} = c\middle|x^{(i)}\right)$$

- Assume you know the parameters, estimate the labels
- We use *soft assignment*: a point can be assigned to y = 1 with probability 0.9 and to y = 0 with probability 0.1

M-step

$$\widehat{count}\left(x_j = 1, y = c\right) = E_{\theta,\pi}\left[\sum_{i|x_j^{(i)}=1} I\left[y^{(i)} = c\right]|x^{(i)}\right] = \sum_{i|x_j^{(i)}=1} E_{\theta,\pi}\left[I\left[y^{(i)} = c\right]|x^{(i)}\right]$$

$$\widehat{count}\,(y = c) = E_{\theta,\pi}\left[\sum_i I\left[y^{(i)} = c\right]|x^{(i)}\right]$$

Re-estimate the θ and π using the new counts

- Compute the counts for each class and feature, assuming that the soft assignments from the E-step are correct
- Re-estimate θ and π

The EM Algorithm: Summary

- Initialiaze π and θ
- Repeat
  - E-step: compute soft assignments for each training sample
  - M-step: re-estimate π and θ based on the new soft assignments

Why does it work?

- Intuitively, the E-step computes the best assignments under the current π and θ
- The M-step computes the best π and θ given the current assignments
- It can be shown* that the EM algorithm optimizes a lower bound on the marginal probability of the data

*But we aren't doing it in this class

Probability of the data

$$P_{\pi,\theta}(x) = \prod_i P_{\pi,\theta}\left(x^{(i)}\right) = \prod_i \sum_y P_{\pi,\theta}(x^{(i)}, y)$$

- Finding π and θ that maximize the probability of the data means finding a model for which the data we observe is likely

Interpreting π and θ

- We don’t know the names of labels, but for each “anonymous” label, we obtain the probability of each keyword appearing

Sample results

- $\theta_A = 0.6, \theta_B = 0.4$
- $P(password|A) = 0.5, P(send|A) = 0.6, P(paper|A) = 0.1, P(password|B) = 0.1, P(send|B) = 0.6, P(paper|B) = 0.3$
- Interpretation: label A means "spam", label B means "not spam"

The EM Algorithm in General

The entire dataset
$x^{(1)}, x^{(2)}, \ldots$

- We observe the data x, and have latent (unobserved) data y. For (unknown) parameters θ, we have the distribution
$$P(x, y|\theta)$$

All unknown params

- We want to learn θ using Maximum Likelihood: find the θ for which $P(x|\theta) = \sum_y P(x, y|\theta)$ is maximized
- If we know y, it's easy to find θ using Maximum likelihood
- If we know θ, it's easy to find $P(y|x, \theta)$