Machine Learning (CSE 446): Expectation-Maximization

Noah Smith

University of Washington
nasmith@cs.washington.edu

December 4, 2017

Unsupervised Learning

The training dataset consists only of $\langle \mathbf{x}_n \rangle_{n=1}^{N}$.

Unsupervised Learning

The training dataset consists only of $\langle \mathbf{x}_n \rangle_{n=1}^{N}$.

There might, or might not, be a test set with correct classes y.

Unsupervised Learning

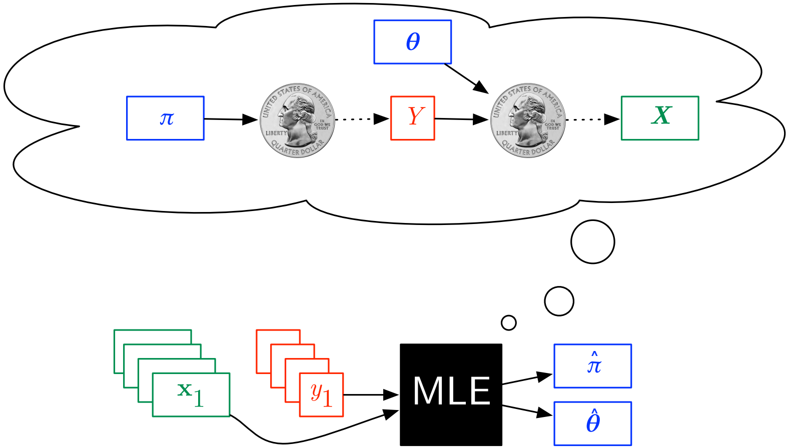
The training dataset consists only of $\langle \mathbf{x}_n \rangle_{n=1}^{N}$.

There might, or might not, be a test set with correct classes y.

Two methods you saw earlier in this course:

- K-means clustering
- Principal components analysis (reduce dimensionality of each instance)

Naïve Bayes Illustrated



Naïve Bayes: Probabilistic Story (All Binary Features)

1. Sample Y according to a Bernoulli distribution where:

$$\begin{aligned} p(Y = +1) &= \pi \\ p(Y = -1) &= 1 - \pi \end{aligned}$$

2. For each feature X_j:
   - Sample X_j according to a Bernoulli distribution where:

$$\begin{aligned} p(X_j = 1 \mid Y = y) &= \theta_{X_j \mid y} \\ p(X_j = 0 \mid Y = y) &= 1 - \theta_{X_j \mid y} \end{aligned}$$

Naïve Bayes: Probabilistic Story (All Binary Features)

1. Sample Y according to a Bernoulli distribution where:

$$\begin{aligned} p(Y = +1) &= \pi \\ p(Y = -1) &= 1 - \pi \end{aligned}$$

2. For each feature X_j:
   - Sample X_j according to a Bernoulli distribution where:

$$\begin{aligned} p(X_j = 1 \mid Y = y) &= \theta_{X_j \mid y} \\ p(X_j = 0 \mid Y = y) &= 1 - \theta_{X_j \mid y} \end{aligned}$$

$1 + 2d$ parameters to estimate: $\pi, \{\theta_{X_j \mid +1}, \theta_{X_j \mid -1}\}_{j=1}^{d}$.

Naïve Bayes: Maximum Likelihood Estimation (All Binary Features)

In general, for a Bernoulli with parameter π, if the observations are $o_1, \ldots, o_N$:

$$\hat{\pi} = \frac{\text{count}(+1)}{\text{count}(+1) + \text{count}(-1)} = \frac{|\{n : o_n = +1\}|}{N}$$

Naïve Bayes: Maximum Likelihood Estimation (All Binary Features)

In general, for a Bernoulli with parameter π, if the observations are $o_1, \ldots, o_N$:

$$\hat{\pi} = \frac{\text{count}(+1)}{\text{count}(+1) + \text{count}(-1)} = \frac{|\{n : o_n = +1\}|}{N}$$

In general, for a conditional Bernoulli for $p(A \mid B)$, if the observations are $(a_1, b_1), \ldots, (a_N, b_N)$:

$$\hat{\theta}_{A|+1} = \frac{\text{count}(A = 1, B = +1)}{\text{count}(B = +1)} = \frac{|\{n : a_n = 1 \wedge b_n = +1\}|}{|\{n : b_n = +1\}|}$$

$$\hat{\theta}_{A|-1} = \frac{\text{count}(A = 1, B = -1)}{\text{count}(B = -1)} = \frac{|\{n : a_n = 1 \wedge b_n = -1\}|}{|\{n : b_n = -1\}|}$$

Naïve Bayes: Maximum Likelihood Estimation (All Binary Features)

In general, for a Bernoulli with parameter π, if the observations are $o_1, \ldots, o_N$:

$$\hat{\pi} = \frac{\text{count}(+1)}{\text{count}(+1) + \text{count}(-1)} = \frac{|\{n : o_n = +1\}|}{N}$$

In general, for a conditional Bernoulli for $p(A \mid B)$, if the observations are $(a_1, b_1), \ldots, (a_N, b_N)$:

$$\hat{\theta}_{A|+1} = \frac{\text{count}(A = 1, B = +1)}{\text{count}(B = +1)} = \frac{|\{n : a_n = 1 \wedge b_n = +1\}|}{|\{n : b_n = +1\}|}$$

$$\hat{\theta}_{A|-1} = \frac{\text{count}(A = 1, B = -1)}{\text{count}(B = -1)} = \frac{|\{n : a_n = 1 \wedge b_n = -1\}|}{|\{n : b_n = -1\}|}$$

So for naïve Bayes' parameters:

- $\hat{\pi} = \dfrac{|\{n : y_n = +1\}|}{N} = \dfrac{N_+}{N}$

Naïve Bayes: Maximum Likelihood Estimation (All Binary Features)

In general, for a Bernoulli with parameter π, if the observations are $o_1, \ldots, o_N$:

$$\hat{\pi} = \frac{\text{count}(+1)}{\text{count}(+1) + \text{count}(-1)} = \frac{|\{n : o_n = +1\}|}{N}$$

In general, for a conditional Bernoulli for $p(A \mid B)$, if the observations are $(a_1, b_1), \ldots, (a_N, b_N)$:

$$\hat{\theta}_{A|+1} = \frac{\text{count}(A = 1, B = +1)}{\text{count}(B = +1)} = \frac{|\{n : a_n = 1 \wedge b_n = +1\}|}{|\{n : b_n = +1\}|}$$

$$\hat{\theta}_{A|-1} = \frac{\text{count}(A = 1, B = -1)}{\text{count}(B = -1)} = \frac{|\{n : a_n = 1 \wedge b_n = -1\}|}{|\{n : b_n = -1\}|}$$

So for naïve Bayes' parameters:

- $\hat{\pi} = \dfrac{|\{n : y_n = +1\}|}{N} = \dfrac{N_+}{N}$
- $\forall j \in \{1, \ldots, d\}, \forall y \in \{-1, +1\}$:

  $\hat{\theta}_{j|y} = \dfrac{|\{n : y_n = y \wedge \mathbf{x}_n[j] = 1\}|}{|\{n : y_n = y\}|} = \dfrac{\text{count}(y, \phi_j = 1)}{\text{count}(y)}$

Back to *Unsupervised*

All of the above assumed that your data included labels. Without labels, you can't estimate parameters.

Back to *Unsupervised*

All of the above assumed that your data included labels. Without labels, you can't estimate parameters.

Or can you?

Back to *Unsupervised*

All of the above assumed that your data included labels. Without labels, you can't estimate parameters.

Or can you?

Interesting insight: if you already had the parameters, you could guess the labels.

Guessing the Labels (Version 1)

Suppose we have the parameters, π and all $\theta_{j|y}$.

$$p_{\pi,\boldsymbol{\theta}}(+1, \mathbf{x}) = \pi \cdot \prod_{j=1}^{d} \theta_{j|+1}^{\mathbf{x}[j]} \cdot (1 - \theta_{j|+1})^{1-\mathbf{x}[j]}$$

$$p_{\pi,\boldsymbol{\theta}}(-1, \mathbf{x}) = (1 - \pi) \cdot \prod_{j=1}^{d} \theta_{j|-1}^{\mathbf{x}[j]} \cdot (1 - \theta_{j|-1})^{1-\mathbf{x}[j]}$$

$$\hat{y} = \begin{cases} +1 & \text{if } p_{\pi,\boldsymbol{\theta}}(+1, \mathbf{x}) > p_{\pi,\boldsymbol{\theta}}(-1, \mathbf{x}) \\ -1 & \text{otherwise} \end{cases}$$

Guessing the Labels (Version 2)

Suppose we have the parameters, π and all $\theta_{j|y}$.

$$
\begin{aligned}
p_{\pi,\boldsymbol{\theta}}(+1 \mid \mathbf{x}) &= \frac{p_{\pi,\boldsymbol{\theta}}(+1, \mathbf{x})}{p_{\pi,\boldsymbol{\theta}}(\mathbf{x})} \\
&= \frac{\pi \cdot \prod_{j=1}^{d} \theta_{j|+1}^{\mathbf{x}[j]} \cdot (1 - \theta_{j|+1})^{1-\mathbf{x}[j]}}{p_{\pi,\boldsymbol{\theta}}(\mathbf{x})} \\
&= \frac{\pi \cdot \prod_{j=1}^{d} \theta_{j|+1}^{\mathbf{x}[j]} \cdot (1 - \theta_{j|+1})^{1-\mathbf{x}[j]}}{\left(\pi \cdot \prod_{j=1}^{d} \theta_{j|+1}^{\mathbf{x}[j]} \cdot (1 - \theta_{j|+1})^{1-\mathbf{x}[j]}\right) + \left((1-\pi) \cdot \prod_{j=1}^{d} \theta_{j|-1}^{\mathbf{x}[j]} \cdot (1 - \theta_{j|-1})^{1-\mathbf{x}[j]}\right)}
\end{aligned}
$$

Guessing the Labels (Version 2)

Suppose we have the parameters, π and all $\theta_{j|y}$.

$$p_{\pi,\boldsymbol{\theta}}(+1 \mid \mathbf{x})$$

Count $\mathbf{x}_n$ *fractionally* as a positive example with "soft count" $p_{\pi,\boldsymbol{\theta}}(+1 \mid \mathbf{x}_n)$, and as a negative example with soft count $p_{\pi,\boldsymbol{\theta}}(-1 \mid \mathbf{x}_n)$.

Guessing the Labels (Version 2)

Suppose we have the parameters, π and all $\theta_{j|y}$.

$$p_{\pi,\boldsymbol{\theta}}(+1 \mid \mathbf{x})$$

Count $\mathbf{x}_n$ *fractionally* as a positive example with "soft count" $p_{\pi,\boldsymbol{\theta}}(+1 \mid \mathbf{x}_n)$, and as a negative example with soft count $p_{\pi,\boldsymbol{\theta}}(-1 \mid \mathbf{x}_n)$.

$$\tilde{N}_+ = \sum_{n=1}^{N} p_{\pi,\boldsymbol{\theta}}(+1 \mid \mathbf{x}_n)$$
$$\tilde{N}_- = \sum_{n=1}^{N} p_{\pi,\boldsymbol{\theta}}(-1 \mid \mathbf{x}_n)$$

Chicken/Egg

If only we had the labels we could estimate the parameters (MLE).

If only we had the parameters, we could guess the labels (previous slides showed two ways to do it, “hard” and “soft”).

Chicken/Egg

If only we had the labels we could estimate the parameters (MLE).
(Really we only need counts of "events" like $\boxed{Y = +1}$ or $\boxed{Y = -1 \text{ and } \phi_j(\boldsymbol{X}) = 0)}$.)

If only we had the parameters, we could guess the labels (previous slides showed two ways to do it, "hard" and "soft").

A Bit of Notation and Terminology

- A **model family**, denoted $\mathcal{M}$, is the specification of our probabilistic model *except* for the parameter values
  - E.g., “naïve Bayes with these particular d binary features.”
- General notation for the entire set of parameters that you’d need to estimate to use $\mathcal{M}$: $\boldsymbol{\Theta}$
- Probability of some r.v. R under model family $\mathcal{M}$ with parameters $\boldsymbol{\Theta}$: $p(R \mid \mathcal{M}(\boldsymbol{\Theta}))$
- Events: countable things that can happen in the probabilistic story, like $\boxed{Y = +1}$ or $\boxed{Y = -1 \text{ and } \phi_j(\boldsymbol{X}) = 0)}$

Expectation Maximization

```
Data: $D = \langle \mathbf{x}_n \rangle_{n=1}^{N}$, probabilistic model family $\mathcal{M}$, number of epochs $E$, initial
      parameters $\mathbf{\Theta}^{(0)}$
Result: parameters $\mathbf{\Theta}$
for $e \in \{1, \ldots, E\}$ do
    # E step
    for $n \in \{1, \ldots, N\}$ do
        calculate "soft counts" under the assumption that $p(Y_n \mid \mathcal{M}(\mathbf{\Theta}^{(e-1)}))$;
    end
    # M step
    $\mathbf{\Theta}^{(e)} \leftarrow \mathrm{argmax}_{\mathbf{\Theta}}\, p(\text{soft counts} \mid \mathcal{M}(\mathbf{\Theta}))$ # (MLE);
end
return $\mathbf{\Theta}^{(E)}$;
```

Algorithm 1: Expectation-Maximization, a general pattern for probabilistic learning (i.e., estimating parameters of a probabilistic model) with incomplete data.

The following slides were added after lecture.

Theory?

It's possible to show that EM is iteratively improving the log-likelihood of the training data, so in some sense it is trying to find:

$$\operatorname*{argmax}_{\pi,\boldsymbol{\theta}} \prod_{n=1}^{N} p_{\pi,\boldsymbol{\theta}}(\mathbf{x}_n) = \operatorname*{argmax}_{\pi,\boldsymbol{\theta}} \prod_{n=1}^{N} \sum_{y} p_{\pi,\boldsymbol{\theta}}(\mathbf{x}_n, y)$$

This is the *marginal* probability of the observed part of the data.

The other parts (here, the random variable Y), are called *hidden* or *latent* variables.

In general, the above objective is not concave, so the best you can hope for is a local optimum.

Variations on EM

- EM is usually taught first for Gaussian mixture models; you can apply it to any incomplete data setting where you have a probabilistic model family $\mathcal{M}$.
- You can start with an M step, if you have a good way to initialize the soft counts instead of parameters.
- EM with a mix of labeled and unlabeled data (hard counts for labeled examples, soft counts for unlabeled ones).
- EM is commonly used with more complex Y (multiclass, or structured prediction).
- **Self-training**: use hard counts instead of soft counts. Does not require a probabilistic learner!

Self Training

```
**Data**: $D = \langle \mathbf{x}_n \rangle_{n=1}^{N}$, supervised learning algorithm $\mathcal{A}$, number of epochs $E$, initial
        classifier $f^{(0)}$
**Result**: classifier $f$
for $e \in \{1, \ldots, E\}$ do
    # E-like step
    for $n \in \{1, \ldots, N\}$ do
        $\hat{y}_n^{(e)} \leftarrow f^{(e-1)}(\mathbf{x}_n)$
    end
    # M-like step
    $f^{(e)} \leftarrow \mathcal{A}\left(\langle(\mathbf{x}_n, \hat{y}_n^{(e)})\rangle_{n=1}^{N}\right)$;
end
return $f^{(E)}$;
```

Algorithm 2: SelfTraining, a general pattern for unsupervised learning.

Practical Advice

When EM succeeds, it seems to be because:

- The model $\mathcal{M}$ is a reasonable reflection of the real-world data-generating process.
- The parameters or soft counts are well-initialized. In some research areas, EM iterations are applied to a series of models $\mathcal{M}_1, \mathcal{M}_2, \ldots$, and each $\mathcal{M}_t$ is used to calculate soft counts for the next $\mathcal{M}_{t+1}$.

It's wise to run EM with multiple random initializations; take the solution that has the largest marginal likelihood of the training data (or the development data).