Week 5: Machine Learning

1. Introduction

1.1. Background

Why we need learning?

- **No model available**: Human designers can not provide models for all possible situations an intelligent agent may encounter
- **Adaptivity:** environment may change over time
- Human don't understand the task well enough, **not possible to manually program**

1.2. Different Ways

1.2.1 Idealistic

- maintain a **belief** over true way the world works **(possible hypotheses)**
- **statistical learning**, always use **Bayes' rule**

1.2.2. Pragmatic

- anything that improves performance
- always use **optimization**

1.3. Idealistic (Bayesian) Perspective

1.3.1. Basic Ideas

- maintain **belief** over how the world works: hypotheses H
  - (**learning**)using bayes rule to update its beliefs

$$P(H|d) = P(d|H)P(H)/P(d)$$

- Then **act** in a bayesian way:

$$V(a) = \sum_h P(h|d)u(a|h)$$

1.3.2. Problems

How to specify the class of hypothese?

- World is complex, need a huge class H
- Limit the class to allow for tractable inference, but the ture model may not in H
- So there are always two **tradeoff**
  - expressiveness of a hypothesis space and the complexity of findindg a good hypothesis with that space

1.4. Pragmatic Perspective

From pragmatic perspective, learning is **optimization.**

There are always two ways:

- **end-to-end learning:**
  - parametrize ‘actions’ using some parameters θ
  - directly optimize V(θ)
- other typical approach, for example **maximum likelihood**

1.5. Classification

Supervised Learning

General Process:

- bag of **training data** $d = < x_i, y_i >_{i=1...N}$
- assumption: labels generated by **‘true’ function** $y = f(x)$
- goal: find hypothesis $h(x) \approx f(x)$

Instantiations:

regression, classification

Unsupervised Learning

- Learn patterns in the input **without explicit feedback (no labels)**
- Most common task is **clustering**

Semi-supervised Learning

- large bag of data, only **a few are labeled**

Active-Learning

- large bag of data, only **a few are labeled**
- what point should we ask an annotator to label

Reinforcement Learning

Learns from a series of reinforcements: **rewards or punishments**

1.6. Generalization

How do wen know $h \approx f$

Approaches

- use theorems: computational/statistical learning theory
    - For example Error
- use experiments:
    - on a new set of data: test data (test data is **never use**d, also not for model selection)

overfitting

2. Maximum Likelihood

2.1. Basic Idea

- instead of computing posterior $P(H|d)$, we optimize

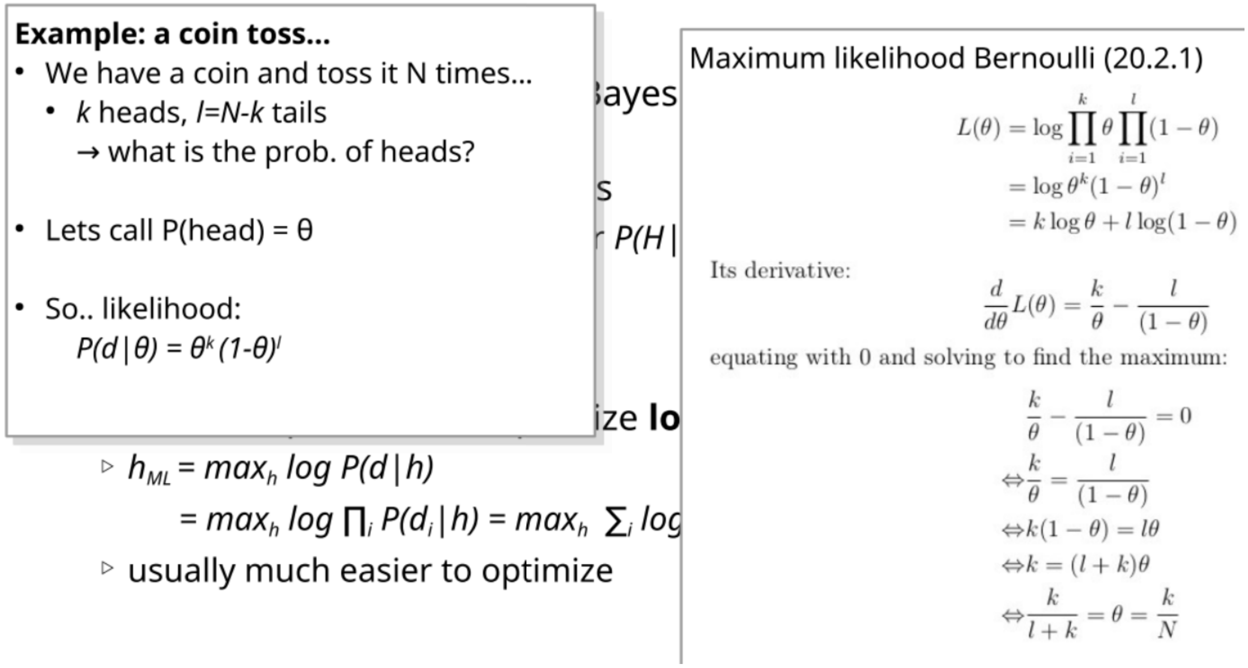
$$h_{ML} = \max_h P(d|h)$$

- to do this optimization, we usually use log likelihood

$$L(h) = logP(d|h)$$

$$\begin{aligned} h_{ML} &= \max_h \log P(d \mid h) \\ &= \max_h \log \Pi_i P\left(d_i \mid h\right) = \max_h \sum_i \log P\left(d_i \mid h\right) \end{aligned}$$

2.2. Example



2.3. Comparison between Maximum Likelihood and Bayesian

Bayesian learning:

- computing posterior $P(H|d)$
  - uses prior information $P(h)$
- use in weighted manner to select best action: $V(a) = \Sigma_h P(h \mid d) u(a, h)$

Maximum likelihood (ML)

- $h_{ML} = \max_h P(d|h)$
- select action according $V(a) = u(a, h_{ML})$
- prone to **"overfitting"**

Maximum a posteriori(MAP):

ML + priors

- $h_{MAP} = max_h\ P(d|h)\ P(h)$
- can still overfit

2.4. Example: Learning Bayesian Network with Maximum Likelihood

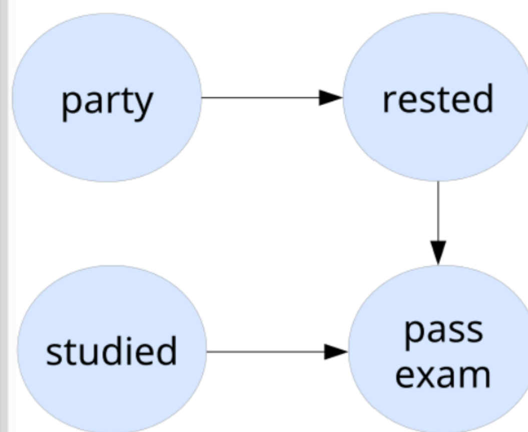
$$L(\theta) = \log P(d;\theta)$$

$$= \log \Pi_i\ P(\langle party_i, rested_i, studied_i, pass_i \rangle ; \theta)$$

$$= \Sigma_i \log P(\langle party_i, rested_i, studied_i, pass_i \rangle ; \theta)$$

$$= \Sigma_i \log P(party_i; \theta)\ P(rested_i | party_i; \theta)\ P(studied_i; \theta) P(pass_i | studied_i, rested_i ; \theta)$$

$$= \Sigma_i \log P(party_i; \theta) + \Sigma_i \log P(rested_i | party_i; \theta) + \Sigma_i \log P(studied_i; \theta) + \Sigma_i \log P(pass_i | studied_i, rested_i ; \theta)$$



3. Learning with hidden variables

3.1. Necessity

Hiddden Variables can help us to eliminate the number of parameters

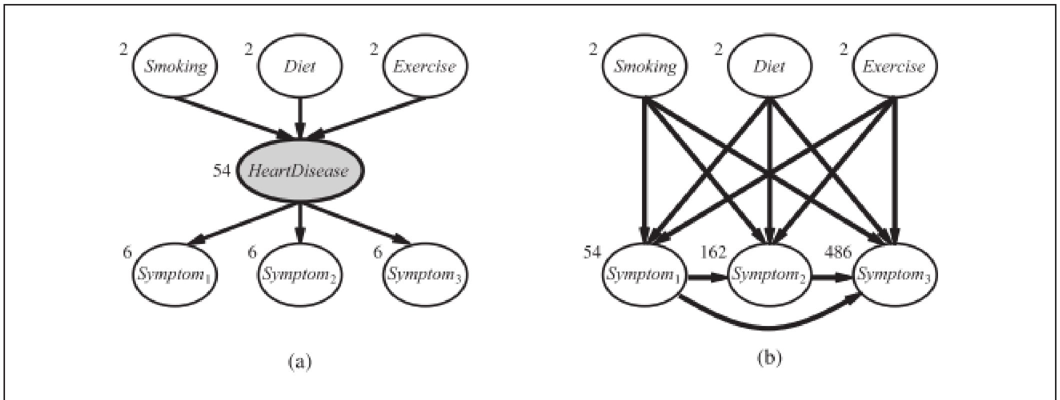


Figure 20.10 (a) A simple diagnostic network for heart disease, which is assumed to be a hidden variable. Each variable has three possible values and is labeled with the number of independent parameters in its conditional distribution; the total number is 78. (b) The equivalent network with *HeartDisease* removed. Note that the symptom variables are no longer conditionally independent given their parents. This network requires 708 parameters.

3.2. Problems

But we don't know the parameter of the hidden variables

3.3. EM algorithm

Intuition

learn better parameters given estimated variables

Basic Idea

$$\theta^{(k+1)} = \arg\max_{\theta} \sum_{\boldsymbol{z}} P\left(\boldsymbol{Z} = \boldsymbol{z} \mid \boldsymbol{x}, \theta^{(k)}\right) L(\boldsymbol{x}, \boldsymbol{Z} = z \mid \theta)$$

where:

- $\theta^{(k+1)}$ is the new parameter vector
- $\boldsymbol{z}$ is the vector of values for latent variables $\boldsymbol{Z}$
- $\boldsymbol{x}$ is the value of observed variables
- $P(\boldsymbol{Z} = \boldsymbol{z} | \boldsymbol{x}, \theta^{(k)})$ the 'estimation' of the latent variables given $\boldsymbol{x}, \theta^{(k)}$
- $L(\boldsymbol{x}, \boldsymbol{Z} = \boldsymbol{z} | \theta)$ the log likelihood:

$$L(\boldsymbol{x}, \boldsymbol{Z} = \boldsymbol{z} | \theta) = \log P(\boldsymbol{x}, \boldsymbol{Z} = \boldsymbol{z} | \theta)$$

Two Steps

1. **E-step**, where 'E' stands for *expectation.* Here the summation over $\boldsymbol{z}$ is performed to compute the expectation. Note that, in order to accomplish this, it needs to compute, or *estimate,* the posterior $P(\boldsymbol{Z}=\boldsymbol{z}|\boldsymbol{x},\theta^{(k)})$.
2. **M-step**. Which performs the maximization over parameters θ.

1. E-step:
   estimate $p_{ij}=P(C=i|x_j)$using current cluster parameters
2. M-step:

   update cluster parameters using p_{ij}

4. Two examples of EM algorithm

4.1. EM for the Bayesian Network

- Due to hidden variable, directly optimizing log likelihood is hard:

$$
\begin{aligned}
L(\boldsymbol{x}|\theta) &= \log \Pr(\boldsymbol{x}|\theta) = \log \sum_{\boldsymbol{z}} \Pr(\boldsymbol{x},\boldsymbol{Z}=\boldsymbol{z}|\theta) \\
&= \log \sum_{\boldsymbol{z}} \prod_{i=1}^{N} \Pr(x_i, Z_i = z_i|\theta) \\
&= \log \sum_{\boldsymbol{z}} \prod_{i=1}^{N} \Pr(z_i|\theta_B) \Pr(flavor_i|z_i,\theta_{Fz_i}) \Pr(wrapper_i|z_i,\theta_{Wz_i}) \Pr(hole_i|z_i,\theta_{Hz_i})
\end{aligned}
$$

- With EM algorithm

$$\theta^{(k+1)} = \arg\max_{\theta} \sum_{\boldsymbol{z}} P(\boldsymbol{z}|\boldsymbol{x},\theta^{(k)})L(\boldsymbol{x},\boldsymbol{z}|\theta)$$

which uses the *full (or 'completed') log-likelihood:*

$$\begin{aligned}
L(\boldsymbol{x},\boldsymbol{z}|\theta) &= \log \Pr(\boldsymbol{x},\boldsymbol{z}|\theta) = \log \prod_{i=1}^{N} \Pr(x_i,z_i|\theta) \\
&= \sum_{i=1}^{N} \log \Pr(x_i,z_i|\theta) \\
&= \sum_{i=1}^{N} \log \Pr(z_i|\theta_B) + \sum_{i=1}^{N} \log \Pr(flavor_i|z_i,\theta_{Fz_i}) \\
&\quad + \sum_{i=1}^{N} \log \Pr(wrapper_i|z_i,\theta_{Wz_i}) + \sum_{i=1}^{N} \log \Pr(hole_i|z_i,\theta_{Hz_i})
\end{aligned}$$

$$\begin{aligned}
\sum_{i=1}^{N} \log \Pr(z_i|\theta_B) &= \sum_{i \text{ s.t. } z_i=1} \log \Pr(z_i|\theta_B) + \sum_{i \text{ s.t. } z_i=2} \log \Pr(z_i|\theta_B) \\
&= \sum_{i \text{ s.t. } z_i=1} \log \theta_B + \sum_{i \text{ s.t. } z_i=2} \log(1-\theta_B) \\
&= N_1 \log \theta_B + N_2 \log(1-\theta_B) \\
&= N_1 \log \theta_B + (N - N_1) \log(1-\theta_B)
\end{aligned}$$

Where $N_1 = N(bag = 1|\boldsymbol{z})$.

If $\boldsymbol{z}$ was correct, this would then lead (by taking derivative and setting to 0) to

$$\theta_B \leftarrow \frac{N_1}{N}$$

$$
\begin{aligned}
&\sum_{\boldsymbol{z}} P(\boldsymbol{z}|\boldsymbol{x},\theta^{(k)})\left[N_1 \log \theta_B + (N - N_1)\log(1-\theta_B)\right] \\
&= \sum_n P(N_1 = n|\boldsymbol{x},\theta^{(k)})\left[n \log \theta_B + (N-n)\log(1-\theta_B)\right] \\
&= \sum_n P(N_1 = n|\boldsymbol{x},\theta^{(k)}) n \log \theta_B + \sum_n P(N_1 = n|\boldsymbol{x},\theta^{(k)})(N-n)\log(1-\theta_B) \\
&= \log \theta_B \cdot \sum_n P(N_1 = n|\boldsymbol{x},\theta^{(k)}) n + \log(1-\theta_B) \cdot (N - \sum_n P(N_1 = n|\boldsymbol{x},\theta^{(k)}) - n) \\
&= \log \theta_B \cdot \hat{N}(Bag = 1) + \log(1-\theta_B) \cdot (N - \hat{N}(Bag = 1))
\end{aligned}
$$

with $\hat{N}(Bag = 1)$ is the *expected* counts for bag 1.

- Finally, this leads (by taking derivative and setting to 0) to

$$\theta_B^{(k+1)} \leftarrow \frac{\hat{N}(Bag = 1)}{N}$$

4.2. EM for HMMs