Graphical Models

introduction to learning

Siamak Ravanbakhsh

Winter 2018

Learning objectives

different goals of learning a graphical model
effect of goals on the learning setup

Where does a graphical model come from?

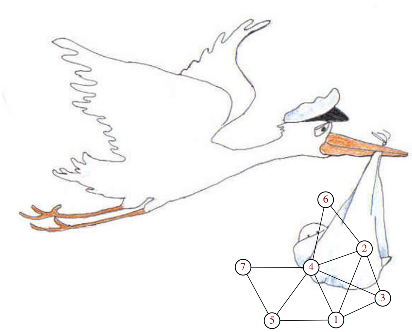


image: http://blog.londolozi.com/

Where does a graphical model come from?

- designed by **domain experts**:
  - more suitable for directed models
    - cond. probabilities are more intuitive than unnormalized factors
    - no need to estimate the partition function

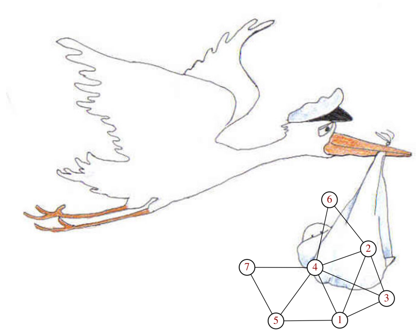
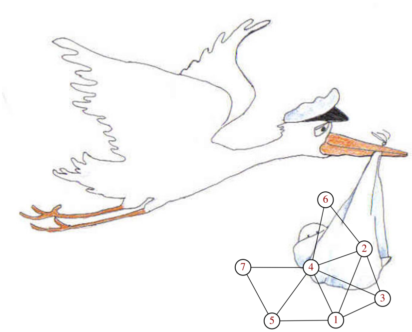


image: http://blog.londolozi.com/

Where does a graphical model come from?



- designed by **domain experts**:
  - more suitable for directed models
    - cond. probabilities are more intuitive than unnormalized factors
    - no need to estimate the partition function
- **learning** from data:
  - fixed structure:
    - easy for directed models
  - unknown structure
  - fully or partially observed data, hidden variables

image: http://blog.londolozi.com/

Goals of learning: density estimation

- **assumption**: data is IID sample from a P^*

$$\mathcal{D} = \{X^{(1)}, \ldots, X^{(M)}\} \quad X^{(m)} \sim P^*$$

empirical distribution: $P_{\mathcal{D}}(x) = \frac{1}{|\mathcal{D}|}\mathbb{I}(x \in \mathcal{D})$

Goals of learning: density estimation

- **assumption**: data is IID sample from a P^*

$$\mathcal{D} = \{X^{(1)}, \ldots, X^{(M)}\} \quad X^{(m)} \sim P^*$$

empirical distribution: $P_{\mathcal{D}}(x) = \frac{1}{|\mathcal{D}|}\mathbb{I}(x \in \mathcal{D})$

- **objective**: learn a $\hat{P}$ close to P^*

$$\hat{P} = \arg\min_P D_{KL}(P^* \| P)$$

Goals of learning: density estimation

- **assumption**: data is IID sample from a P^*

$$\mathcal{D} = \{X^{(1)}, \ldots, X^{(M)}\} \quad X^{(m)} \sim P^*$$

empirical distribution: $P_{\mathcal{D}}(x) = \frac{1}{|\mathcal{D}|}\mathbb{I}(x \in \mathcal{D})$

- **objective**: learn a $\hat{P}$ close to P^*

$$\hat{P} = \arg\min_P D_{KL}(P^* \| P) = \mathbb{E}_{P^*}[\log P^*] - \mathbb{E}_{P^*}[\log P]$$

Goals of learning: density estimation

- **assumption**: data is IID sample from a P^*

$$\mathcal{D} = \{X^{(1)}, \ldots, X^{(M)}\} \quad X^{(m)} \sim P^*$$

empirical distribution: $P_{\mathcal{D}}(x) = \frac{1}{|\mathcal{D}|}\mathbb{I}(x \in \mathcal{D})$

- **objective**: learn a $\hat{P}$ close to P^*

$$\hat{P} = \arg\min_P D_{KL}(P^* \| P) = \mathbb{E}_{P^*}[\log P^*] - \mathbb{E}_{P^*}[\log P]$$

negative Entropy of P* (does not depend on P)

Goals of learning: density estimation

- **assumption**: data is IID sample from a P^*

$$\mathcal{D} = \{X^{(1)}, \ldots, X^{(M)}\} \quad X^{(m)} \sim P^*$$

empirical distribution: $P_{\mathcal{D}}(x) = \frac{1}{|\mathcal{D}|}\mathbb{I}(x \in \mathcal{D})$

- **objective**: learn a $\hat{P}$ close to P^*

$$\hat{P} = \arg\min_P D_{KL}(P^* \| P) = \mathbb{E}_{P^*}[\log P^*] - \mathbb{E}_{P^*}[\log P]$$

negative Entropy of P* (does not depend on P)

substitute P^* with $P_{\mathcal{D}}$: $\hat{P} = \arg\max_P \sum_{x \in \mathcal{D}} \log P(x)$

how to compare two log-likelihood values?

Goals of learning: density estimation

- **assumption**: data is IID sample from a P^*

$$\mathcal{D} = \{X^{(1)}, \ldots, X^{(M)}\} \quad X^{(m)} \sim P^*$$

empirical distribution: $P_{\mathcal{D}}(x) = \frac{1}{|\mathcal{D}|}\mathbb{I}(x \in \mathcal{D})$

- **objective**: learn a $\hat{P}$ close to P^*

$$\hat{P} = \arg\min_P D_{KL}(P^* \| P) = \mathbb{E}_{P^*}[\log P^*] - \mathbb{E}_{P^*}[\log P]$$

negative Entropy of P* (does not depend on P)

substitute P^* with $P_{\mathcal{D}}$: $\hat{P} = \arg\max_P \sum_{x \in \mathcal{D}} \log P(x)$

log-likelihood

how to compare two log-likelihood values?

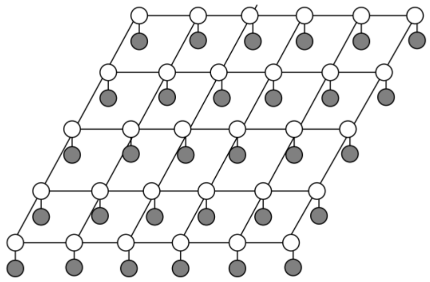
Goals of learning: prediction

- given $\mathcal{D} = \{(X^{(m)}, Y^{(m)})\}$

  interested in learning $\hat{P}(X \mid Y)$

  the output in our prediction is structured

  making prediction: $\hat{x}(y) = \arg\max_x \hat{P}(x \mid y)$



e.g. in image segmentation

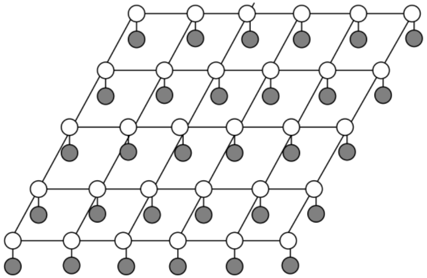
Goals of learning: prediction

- given $\mathcal{D} = \{(X^{(m)}, Y^{(m)})\}$

  interested in learning $\hat{P}(X \mid Y)$

  the output in our prediction is structured

  making prediction: $\hat{x}(y) = \arg\max_x \hat{P}(x \mid y)$



e.g. in image segmentation

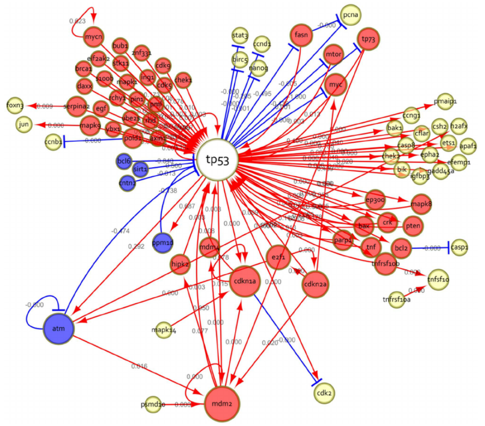
- error measures:
  - 0/1 loss (unforgiving): $\mathbb{E}_{(X,Y)\sim P^*} \mathbb{I}(X = \hat{X}(Y))$
  - Hamming loss: $\mathbb{E}_{(X,Y)\sim P^*} \sum_i \mathbb{I}(X_i = \hat{X}(Y)_i)$
  - conditional log-likelihood: $\mathbb{E}_{(X,Y)\sim P^*} \log \hat{P}(X \mid Y)$
    - takes prediction uncertainty into account

Goals of learning: knowledge discovery

given $\mathcal{D} = \{(X^{(m)})\}$

interested in learning $\mathcal{G}$ or $\mathcal{H}$

finding conditional independencies or causal relationships



E.g. in gene regulatory network

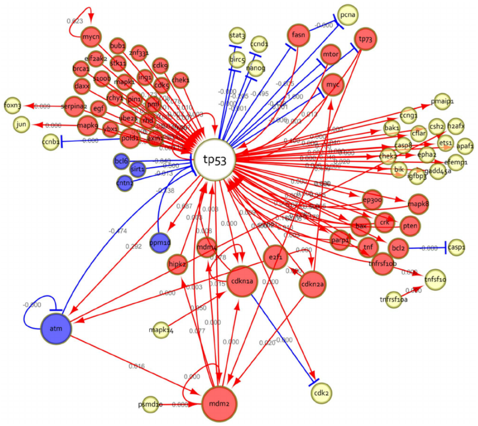
image credit: Chen et al., 2014

Goals of learning: knowledge discovery

given $\mathcal{D} = \{(X^{(m)})\}$

interested in learning $\mathcal{G}$ or $\mathcal{H}$

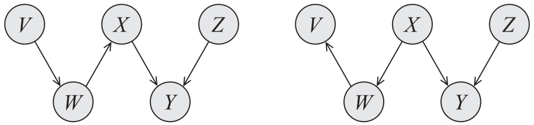
finding conditional independencies or causal relationships



E.g. in gene regulatory network

not always uniquely **identifiable**

Recall two DAGs are I-equivalent if $\mathcal{I}(\mathcal{G}) = \mathcal{I}(\mathcal{G}')$



- same undirected skeleton
- same immoralities

image credit: Chen et al., 2014

bias-variance trade-off

learning *ideally* minimizes some risk (expected loss) $\mathbb{E}_{X \sim P^*}[loss(X)]$

in reality we use empirical risk $\mathbb{E}_{x \in \mathcal{D}}[loss(x)]$

image: http://ipython-books.github.io

bias-variance trade-off

learning *ideally* minimizes some risk (expected loss) $\mathbb{E}_{X \sim P^*}[loss(X)]$
in reality we use empirical risk $\mathbb{E}_{x \in \mathcal{D}}[loss(x)]$

if our model is expressive we can overfit

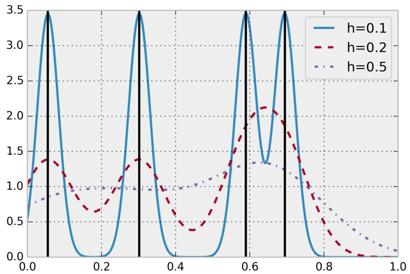
high variance

low *empirical* risk does not translate to low risk

our model does not generalize to samples outside $\mathcal{D}$

as measured by a validation set

different choices of $\mathcal{D} \sim P^*$ produce very different models $\hat{P}$



overfitting in density estimation

image: http://ipython-books.github.io

bias-variance trade-off

learning *ideally* minimizes some risk (expected loss) $\mathbb{E}_{X \sim P^*}[loss(X)]$

in reality we use empirical risk $\mathbb{E}_{x \in \mathcal{D}}[loss(x)]$

if our model is expressive we can overfit

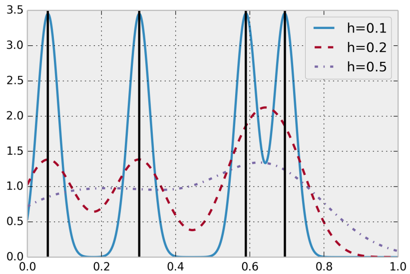
high variance

low *empirical* risk does not translate to low risk

our model does not generalize to samples outside $\mathcal{D}$

as measured by a validation set

different choices of $\mathcal{D} \sim P^*$ produce very different models $\hat{P}$



overfitting in density estimation

high bias

simple models cannot fit the data

- the model has a bias even for large $\mathcal{D}$

image: http://ipython-books.github.io

bias-variance trade-off

learning *ideally* minimizes some risk (expected loss) $\mathbb{E}_{X \sim P^*}[loss(X)]$
in reality we use empirical risk $\mathbb{E}_{x \in \mathcal{D}}[loss(x)]$

if our model is expressive we can overfit

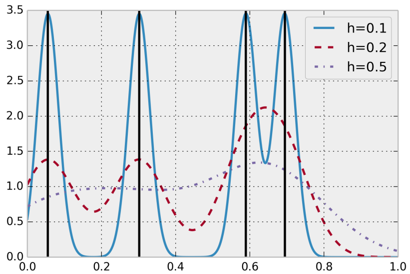
high variance

low *empirical* risk does not translate to low risk

our model does not generalize to samples outside $\mathcal{D}$

as measured by a validation set

different choices of $\mathcal{D} \sim P^*$ produce very different models $\hat{P}$



overfitting in density estimation

a solution: penalize model complexity **regularization**

high bias

simple models cannot fit the data

- the model has a bias even for large $\mathcal{D}$

image: http://ipython-books.github.io

Discreminative vs generative training

if the goal is prediction: $\hat{P}(X \mid Y)$

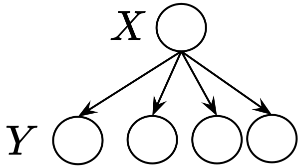
- **Generative:** learn $\hat{P}(X, Y)$ and condition on Y (e.g., MRF)
- **Discriminative:** directly learn $\hat{P}(X \mid Y)$ (e.g., CRF)

Discreminative vs generative training

if the goal is prediction: $\hat{P}(X \mid Y)$

- **Generative:** learn $\hat{P}(X, Y)$ and condition on Y (e.g., MRF)
- **Discriminative:** directly learn $\hat{P}(X \mid Y)$ (e.g., CRF)

Example naive Bayes vs logistic regression



- trained generatively (log-likelihood)
- works better on small datasets (higher bias)
- unnecessary cond. ind. assumptions about Y
- can deal with missing values & learn from unlabeled data

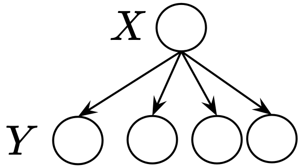
Naive Bayes $P(X \mid Y) \propto P(X)P(Y \mid X)$

Discreminative vs generative training

if the goal is prediction: $\hat{P}(X \mid Y)$

- **Generative:** learn $\hat{P}(X, Y)$ and condition on Y (e.g., MRF)
- **Discriminative:** directly learn $\hat{P}(X \mid Y)$ (e.g., CRF)

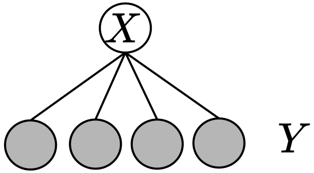
Example naive Bayes vs logistic regression



- trained generatively (log-likelihood)
- works better on small datasets (higher bias)
- unnecessary cond. ind. assumptions about Y
- can deal with missing values & learn from unlabeled data

Naive Bayes $P(X \mid Y) \propto P(X)P(Y \mid X)$

- trained discriminatively (cond. log-likelihood)
- works better on large datasets
- no assumptions about cond. ind. in Y



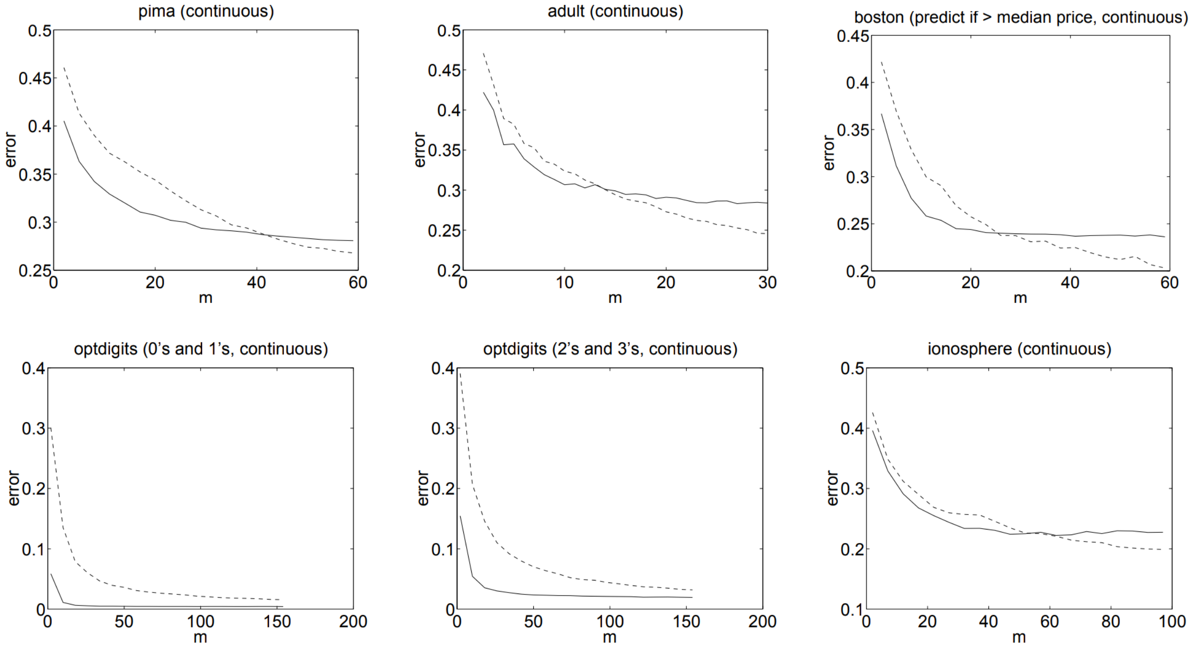
logistic regression $P(X = 1|Y) = \sigma(WY + b)$

Discreminative vs generative training

Example naive Bayes vs logistic regression on UCI dataset

naive Bayes

logistic regression



from: Ng & Jordan 2001

summary

- learning can have different objectives:
  - density estimation
    - calculating P(x)
    - sampling from P (generative modeling)
  - prediction (conditional density estimation)
    - discriminative and generative modeling
  - knowledge discovery
- expressed as empirical risk minimization
  - bias-variance trade-off
  - regularize the model