CS 2750 Machine Learning Lecture 12b

Bayesian belief networks

Milos Hauskrecht <u>milos@cs.pitt.edu</u> 5329 Sennott Square

Density estimation

Data: $D = \{D_1, D_2, ..., D_n\}$ $D_i = \mathbf{x}_i$ a vector of attribute values

Attributes:

- modeled by random variables $\mathbf{X} = \{X_1, X_2, \dots, X_d\}$ with:
 - Continuous values
 - Discrete values
 - E.g. *blood pressure* with numerical values
 - or *chest pain* with discrete values

[no-pain, mild, moderate, strong]

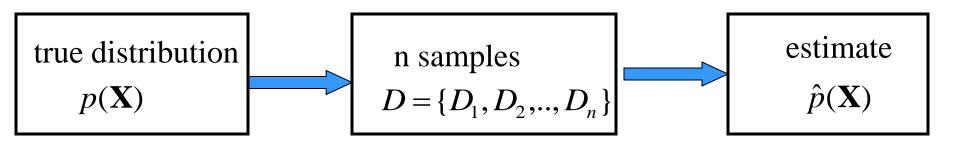
Underlying true probability distribution:

 $p(\mathbf{X})$

Density estimation

Data: $D = \{D_1, D_2, ..., D_n\}$ $D_i = \mathbf{x}_i$ a vector of attribute values

Objective: try to estimate the underlying true probability distribution over variables \mathbf{X} , $p(\mathbf{X})$, using examples in D



Standard (iid) assumptions: Samples

- are independent of each other
- come from the same (identical) distribution (fixed p(X))

Learning via parameter estimation

In this lecture we consider **parametric density estimation Basic settings:**

- A set of random variables $\mathbf{X} = \{X_1, X_2, \dots, X_d\}$
- A model of the distribution over variables in X with parameters Θ : $\hat{p}(\mathbf{X} | \Theta)$

• **Data**
$$D = \{D_1, D_2, ..., D_n\}$$

Objective: find the parameters Θ that explain best the observed data

Parameter estimation

• Maximum likelihood (ML)

maximize $p(D|\Theta,\xi)$

- yields: one set of parameters Θ_{ML}
- the target distribution is approximated as:

 $\hat{p}(\mathbf{X}) = p(\mathbf{X} \mid \boldsymbol{\Theta}_{ML})$

- Bayesian parameter estimation
 - uses the posterior distribution over possible parameters

$$p(\Theta \mid D, \xi) = \frac{p(D \mid \Theta, \xi) p(\Theta \mid \xi)}{p(D \mid \xi)}$$

- Yields: all possible settings of Θ (and their "weights")
- The target distribution is approximated as: $\hat{p}(\mathbf{X}) = p(\mathbf{X} \mid D) = \int p(\mathbf{X} \mid \mathbf{\Theta}) p(\mathbf{\Theta} \mid D \in \mathcal{E})$

$$\hat{p}(\mathbf{X}) = p(\mathbf{X} \mid D) = \int_{\Theta} p(X \mid \Theta) p(\Theta \mid D, \xi) d\Theta$$

Parameter estimation

Other possible criteria:

- Maximum a posteriori probability (MAP)
 - maximize $p(\boldsymbol{\Theta} | D, \xi)$ (mode of the posterior)

 Θ_{MAP}

- Yields: one set of parameters
- Approximation:

$$\hat{p}(\mathbf{X}) = p(\mathbf{X} \mid \boldsymbol{\Theta}_{MAP})$$

- Expected value of the parameter
 - $\hat{\boldsymbol{\Theta}} = E(\boldsymbol{\Theta})$ (mean of the posterior)
 - Expectation taken with regard to posterior $p(\Theta | D, \xi)$
 - Yields: one set of parameters
 - Approximation:

$$\hat{p}(\mathbf{X}) = p(\mathbf{X} \,|\, \hat{\mathbf{\Theta}})$$

Density estimation

- So far we have covered density estimation for "simple" distribution models:
 - Bernoulli
 - Binomial
 - Multinomial
 - Gaussian
 - Poisson

But what if:

- The dimension of $\mathbf{X} = \{X_1, X_2, \dots, X_d\}$ is large
 - Example: patient data
- Compact parametric distributions do not seem to fit the data
 E.g.: multivariate Gaussian may not fit
- We have only a "small" number of examples to do accurate parameter estimates

How to learn complex distributions

How to learn complex multivariate distributions $\hat{p}(\mathbf{X})$ with large number of variables?

One solution:

- Decompose the distribution using conditional independence relations
- Decompose the parameter estimation problem to a set of smaller parameter estimation tasks

Decomposition of distributions under conditional independence assumption is the main idea behind **Bayesian belief networks**

Example

Problem description:

- **Disease:** pneumonia
- Patient symptoms (findings, lab tests):
 - Fever, Cough, Paleness, WBC (white blood cells) count, Chest pain, etc.

Representation of a patient case:

- Symptoms and disease are represented as random variables
 Our objectives:
- Describe a multivariate distribution representing the relations between symptoms and disease
- Design of inference and learning procedures for the multivariate model

Modeling uncertainty with probabilities

- Full joint distribution:
 - Assume $\mathbf{X} = \{X_1, X_2, \dots, X_d\}$ are all random variables that define the domain
 - Full joint: $P(\mathbf{X})$ or $P(X_1, X_2, ..., X_d)$
- **Full joint it is sufficient** to do any type of probabilistic inference:
- Computation of joint probabilities for sets of variables

 $P(X_1, X_2, X_3)$ $P(X_1, X_{10})$

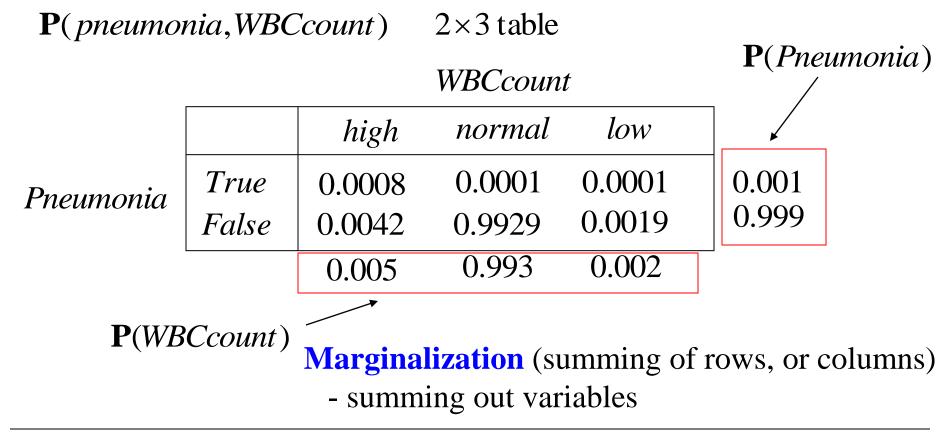
• Computation of conditional probabilities

$$P(X_1 | X_2 = True, X_3 = False)$$

Marginalization

Joint probability distribution (for a set variables)

• Defines probabilities for all possible assignments to values of variables in the set



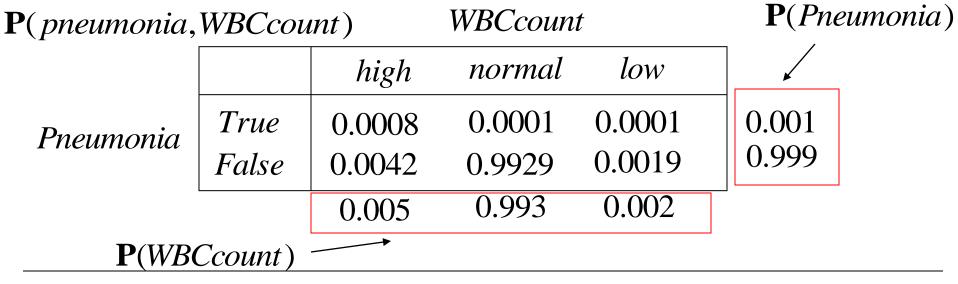
CS 2750 Machine Learning

Variable independence

- The joint distribution over a subset of variables can be always computed from the joint distribution through marginalization
- Not the other way around !!!

- Only exception: when variables are independent

P(A,B) = P(A)P(B)



CS 2750 Machine Learning

Conditional probability

Conditional probability :

• Probability of A given B

$$P(A \mid B) = \frac{P(A, B)}{P(B)}$$

- Conditional probability is defined in terms of joint probabilities
- Joint probabilities can be expressed in terms of conditional probabilities

P(A, B) = P(A | B)P(B) (product rule)

 $P(X_1, X_2, \dots, X_n) = \prod_{i=1}^n P(X_i \mid X_{1, i}, \dots, X_{i-1})$ (chain rule)

Conditional probability – is useful for various probabilistic inferences

P(*Pneumonia* = *True* | *Fever* = *True*, *WBCcount* = *high*, *Cough* = *True*)

Inference

Any query can be computed from the full joint distribution !!!

• **Joint over a subset of variables** is obtained through marginalization

$$P(A = a, C = c) = \sum_{i} \sum_{j} P(A = a, B = b_i, C = c, D = d_j)$$

• **Conditional probability over a set of variables**, given other variables' values is obtained through marginalization and definition of conditionals

$$P(D = d | A = a, C = c) = \frac{P(A = a, C = c, D = d)}{P(A = a, C = c)}$$
$$= \frac{\sum_{i}^{i} P(A = a, B = b_{i}, C = c, D = d)}{\sum_{i}^{i} \sum_{j}^{i} P(A = a, B = b_{i}, C = c, D = d_{j})}$$

CS 2750 Machine Learning

Inference

• Any joint probability can be expressed as a product of conditionals via the **chain rule**.

$$P(X_1, X_2, \dots, X_n) = P(X_n | X_{1,} \dots, X_{n-1}) P(X_{1,} \dots, X_{n-1})$$

= $P(X_n | X_{1,} \dots, X_{n-1}) P(X_{n-1} | X_{1,} \dots, X_{n-2}) P(X_{1,} \dots, X_{n-2})$
= $\prod_{i=1}^n P(X_i | X_{1,} \dots, X_{i-1})$

• It is often easier to define the distribution in terms of conditional probabilities:

- E.g.
$$\mathbf{P}(Fever | Pneumonia = T)$$

 $\mathbf{P}(Fever | Pneumonia = F)$

Modeling uncertainty with probabilities

- **Full joint distribution:** joint distribution over all random variables defining the domain
 - it is sufficient to represent the complete domain and to do any type of probabilistic inferences

Problems:

- Space complexity. To store full joint distribution requires to remember $O(d^n)$ numbers.
 - n number of random variables, d number of values
- Inference complexity. To compute some queries requires $O(d^n)$ steps.
- Acquisition problem. Who is going to define all of the probability entries?

Pneumonia example. Complexities.

- Space complexity.
 - Pneumonia (2 values: T,F), Fever (2: T,F), Cough (2: T,F),
 WBCcount (3: high, normal, low), paleness (2: T,F)
 - Number of assignments: 2*2*2*3*2=48
 - We need to define at least 47 probabilities.
- Time complexity.
 - Assume we need to compute the probability of Pneumonia=T from the full joint

P(Pneumonia = T) =

 $= \sum_{i \in T, F} \sum_{j \in T, F} \sum_{k=h,n,l} \sum_{u \in T, F} P(Fever = i, Cough = j, WBCcount = k, Pale = u)$

– Sum over 2*2*3*2=24 combinations

Bayesian belief networks (BBNs)

Bayesian belief networks.

- Represent the full joint distribution over the variables more compactly with a **smaller number of parameters**.
- Take advantage of **conditional and marginal independences** among random variables
- A and B are independent

P(A,B) = P(A)P(B)

• A and B are conditionally independent given C

$$P(A, B | C) = P(A | C)P(B | C)$$
$$P(A | C, B) = P(A | C)$$