Statistical Methods for Machine Learning II

© by Xia, Tingfeng

2020 winter term

Preface

This document is consist of notes from lectures and the online course notes that I, personally, find interesting/important. You can find the online course notes on the course website here: `https://probmlcourse.github.io/sta414/`

Contents

1 Lecture 2 - Introduction to Probabilistic Models

1.1 Overview

We have a random vector in the form $X = (X_1, \ldots, X_m)$ which can be ***either observed or unobserved***. To approach this in a generative way, we make the so called generative assumption. which is that $X \sim P_{true}(X)$, i.e. there is some true distribution that is behind the scene and our data is from such distribution.

Goal Model a parametric joint distribution $P_\theta(X)$ by learning the parameters. The learning here means we want to find a/the "close"/"best" estimation to our parameter θ. In this course we will investigate the following three problems,

- How to specify the joint, $P_\theta(X)$?
- What does "best"/"close" mean? In some sense we want to find $P_\theta \approx P_{true}$, however P_{true} might also be unknown.
- How to find the best θ? In this course we will generally rely on gradient methods, so ∇_θ...

1.2 Probabilistic Perspective on ML

With this perspective, we can think about common machine learning tasks differently, where random variables represent:

- X: (high dimensional) input data
- C: discrete label
- Y: continuous target

If we assume our knowledge of the joint of the above three, i.e. we know $P(X, C, Y)$, then we can write our familiar tasks in the following way

- ***Regression***:
$$p(Y|X) = \frac{p(X,Y)}{P(X)} = \frac{p(X,Y)}{\int p(X,Y)dY}$$
- ***Classification/Clustering***:
$$p(C|X) = \frac{p(X,C)}{\sum_{C'} p(X,C')}$$

1.2.1 (Example) Classification

Suppose we have data of the form $\mathcal{D} = \{(x,c)_i\}_i$. We assume that they came from a certain true distribution, i.e. $\{(x,c)_i\}_i \sim p(X,C)$. Then, the ultimate goal of the ML problem is converted into finding $p(C|X)$. Using Bayes Rule of total probability, we can expand the distribution of interest into

$$p(C|X) = \frac{p(X,C)}{P(X)} = \frac{p(X,C)}{\sum_{C'} p(X,C')}$$

Output Heuristics After we acquire $p(C|X)$ as above, we are one step away from our goal of output the actual prediction c^*. There are three ways that we can do this, namely

- ***MLE Estimate*** is the most intuitive one, we simply choose

$$c^* = \arg\max_c p(C = c|X)$$

- ***Sample Learnt Dist*** is another approach which produces non-deterministic results, i.e. we sample $c^* \sim p(C|X)$.

- ***Combined*** is usually a safe way of doing this. We output

$$(c^*, p(C = c^*|X)) \iff (\text{result}, \text{how sure?})$$

  As an example, we have a ML algorithm drives a car. In this case, we might want to make decision only when the machine learning model has a certain level of confidence.

1.3 Observed vs Unobserved Random Variables

1.3.1 Supervised Dataset

$$\{x_i, c_i\}_{i=1}^N \sim p(X, C)$$

In such case, the class labels are observed and finding the conditional distribution $p(C|X)$ satisfies the supervised classification problem.

1.3.2 Unsupervised Dataset

$$\{x_i\}_{i=1}^N \sim p(X, C)$$

Still under the generative assumption, where we assume that there is some underlying distribution for our dataset. Further, we assume that the distribution of data is related to the class labels for the data points even though the class labels are never observed. **A common way to refer to an unobserved discrete class label is "cluster"**. However, in this case, our final goal of classification is still $p(C|X)$[1].

1.3.3 Latent Variables

Further, like clusters, introducing assumptions about unobserved variables is a powerful modelling tool. We will make use of this by modelling variables which are never observed in the dataset, called latent or hidden variables. By introducing and modelling latent variables, we will be able to naturally describe and capture abstract features of our input data.

[1] Might be helpful to think of Gaussian Mixture Models

1.4 Operations on Probabilistic Models

- ***Generate Data***: Sample from the model.
- ***Estimate Likelihood***: When all variables are either observed or marginalized, we produce the result which is a single real number that describes the 'probability' of the all variables taking on those specific values.
- ***Inference***: Compute the expected value of some variables given others which are either observed or marginalized.
- ***Learning***: Set the parameters of the joint distribution given some observed data to maximize the probability of the observed data.

1.5 Desiderata of Probabilistic Models

We have two desires for the joint distribution to learn, namely

- The marginal and conditional distribution can be computed efficiently
- The representation of the joint distribution should be compact. This is especially important when we are dealing with joint distributions over many variables.

In general, total joint distribution are too large to specify and would require an insane amount of data to fit even we wanted to. Thus, we need modelling assumptions.

1.5.1 Fully Dependent Factorization (Chain Rule)

Suppose we have sample space defined such that $|T| = 2$, $|W| = 3$, and $|M| = 4$. Then the total joint distribution could be expanding using the chain rule as (*note: not unique*)

$$P_\theta(T, W, M) = P(T)P(W|T)P(M|T, W)$$

which requires[2] $|\theta| = (2-1) + (3-1) \times 2 + (4-1) \times 2 \times 3 = 23$ parameters in total to specify.

1.5.2 Assumptions (Independence)

Introducing assumptions results in

- a less expressive model
- $|\theta|$ (usually, much) smaller

which can be bad sometimes (since the model is less expressive) and is a trade-off that we as modellers have to deal with.

(Example) Fully Independent We assume that $T \perp W \perp M$, then by definition we know, for example, $P(W|T) = P(W)$. Then

$$\begin{aligned} P_\theta(T, W, M) &= P(T)P(W|T)P(M|T, W) \\ &= P(T)P(W)P(M) \end{aligned}$$

which requires only $|\theta| = (2-1) + (3-1) + (4-1) = 6$ to fit.

[2]Here the bars mean "cardinality" rather than vector norm

1.6 Likelihood Function

For some observed data X, the likelihood describes the likeliness of the data under then distribution with parameter θ.

$$L(\theta) = p(X|\theta)$$

In general, we prefer to deal with the log likelihood function, which is defined as

$$\ell(\theta; X) = \log L(\theta) = \log p(X|\theta)$$

1.7 Maximum Likelihood Estimation

The idea is to find

$$\hat{\theta}_{MLE} := \arg\max_{\theta} \ell(\theta; \mathcal{D})$$

In the case of i.i.d, we can re-write as

$$\begin{aligned}
\hat{\theta}_{MLE} &= \arg\max_{\theta} \ell(\theta; \mathcal{D}) \\
&= \arg\max_{\theta} \log \prod_{m} p\left(x^{(m)}|\theta\right) \\
&= \arg\max_{\theta} \sum_{m} \log p\left(x^{(m)}|\theta\right)
\end{aligned}$$

1.8 Sufficient Statistics

(Definition) Statistic A statistic is a possibly vector valued *deterministic* function of a set of random variables.

(Definition) Sufficient Statistic is a statistic that conveys exactly the same information about the data generating process that created the data as the entire data itself.[3] In formal language, Sufficient Statistic ($T(X)$) for X could be defined as

$$T(X^{(1)}) = T(X^{(2)}) \implies L(\theta; X^{(1)}) = L(\theta; X^{(2)}), \quad \forall\theta$$

alternatively, we can define it as

$$P(\theta|T(X)) = P(\theta|X); \quad \text{i.e. data doesn't give further info}$$

1.8.1 Fisher-Neyman Factorization Theorem

If the probability function is $f_\theta(x)$, then T is a sufficient statistic for θ if and only if non-negative functions g and h can be found such that

$$P(\theta|T(X)) = h(x, T(x))g(T(x), \theta)$$

[3] Could also be interpreted as "summarize the data with respect to the likelihood"

1.9 Exponential Family

Factorizes as

$$\begin{aligned} p(x|\eta) &= h(x)\exp\left\{\eta' T(x) - g(\eta)\right\} \\ &= h(x)g(\eta)\exp\left\{\eta' T(x)\right\} \end{aligned}$$

1.9.1 (Example) 1-D Gaussian

$$\begin{aligned} p(x|\theta) = \mathcal{N}(x|\mu,\sigma) &= \frac{1}{\sqrt{2\pi\sigma^2}}\exp\left(-\frac{1}{2\sigma^2}(x-\mu)^2\right) \\ &= \frac{1}{\sqrt{2\pi\sigma^2}}\exp\left(-\frac{1}{2\sigma^2}(x^2 - 2x\mu + \mu^2)\right) \\ &= \underbrace{\frac{1}{\sqrt{2\pi}\sigma}\exp\left(\frac{-\mu^2}{2\sigma^2}\right)}_{\log A(\eta)}\exp\left(\underbrace{\begin{bmatrix}\frac{\mu}{\sigma^2} & \frac{-1}{2\sigma^2}\end{bmatrix}}_{\eta}\underbrace{\begin{bmatrix}x\\ x^2\end{bmatrix}}_{T(X)}\right) \end{aligned}$$

2 Lecture 3 - Directed Graphical Models

2.1 Decision Theory (Utility Theory)

Let a denote action, and a^* be the optimal one. Also use s to denote state and $V(\cdot)$ be the value function. We have, in general

$$a^* = \arg\min_a / \arg\max_a \underbrace{\mathbb{E}_{p(s|a,knowledge)}\left[V(s)\right]}_{u(a) \triangleq \text{ utility of action } a}$$

2.2 Graphical Model Notation

2.2.1 Chain Rule Expansion

Given any joint probability of N random variables, we can expand it as follows

$$p\left(x_{1,\ldots,N}\right) = p\left(x_1\right)p\left(x_2|x_1\right)p\left(x_3|x_2,x_1\right)\ldots p\left(x_n|x_{n-1:1}\right)$$

Formally speaking, in the case of two random variables would simply

$$p(x,y) = p(x|y)p(y)$$

and the general case for N random variables could be written as [4]

$$p\left(\bigcap_{i=1}^{N} x_i\right) = \prod_{j=1}^{N} p\left(x_j \,\middle|\, \bigcap_{k=1}^{j-1} x_k\right)$$

[4] Note: when $k=1$, $p\left(x_k|\cap_{j=1}^{k-1} x_j\right) = p(x_1)$

2.2.2 Graph Representation

(Example) Grouping Variables Consider the model

$$p\left(x_i, x_{\pi_i}\right) = p\left(x_{\pi_i}\right) p\left(x_i | x_{\pi_i}\right)$$

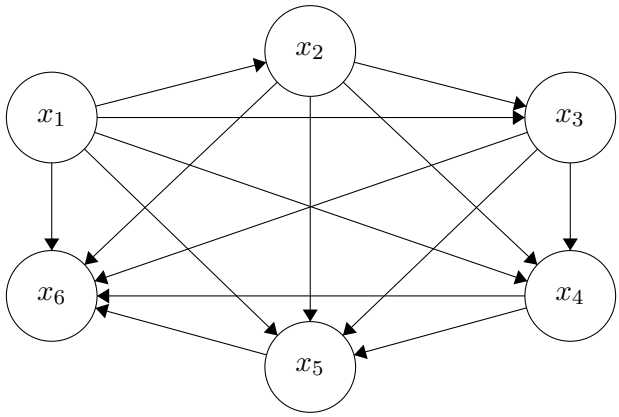
which we can use the following graph to represent:

where

- ***nodes*** represent random variables
- ***arrows*** mean "conditioned on", e.g. "x_i is conditioned on x_{π_i}"

Notice that we can always group the variables together into one bigger variable, so in this example, x_{π_i} might represent a group of variables in stead of just one.

(Example) Fully Dependent 6 Nodes The total expansion of $p(x_{1:6})$ could be represented as



This is the resultant graphical model if we make ***absolutely no assumption*** on the independence. Notice that such model grows exponentially in complexity with respect to the number of parameters considered. We say such model "scales poorly".

2.2.3 Conditional Independence

(Definition) Conditional Independence Let X be the set of nodes in our graph (the random variables of our model), then two sets of variables X_A, X_B are said to be conditionally independent given a third set of variables X_C if and only if either

$$p\left(X_A, X_B | X_C\right) = p\left(X_A | X_C\right) p\left(X_B | X_C\right)$$

or (***#! Important!***)

$$\begin{aligned} & p\left(X_A | X_B, X_C\right) = p\left(X_A | X_C\right) \\ \iff & p\left(X_B | X_A, X_C\right) = p\left(X_B | X_C\right) \end{aligned}$$

and we denote the relation as $(X_A \perp X_B | X_C)$

2.2.4 Plates

In Bayesian methods, we treat parameters as random variables and hence we would like to include them in out graphical model. However, adding a node for each observation is quite cumbersome and thus we introduce plates, which denote replication of random variables.

Nested Plates Plates could be nested, in which case their arrows get duplicated also, ***according to the rule:*** draw an arrow from every copy of the source node to every copy of the destination node.

Crossing Plates Plates can also cross (intersect), in which case the nodes at the intersection have multiple indices and get duplicated a number of times equal to the product of the duplication numbers on all the plates containing them.

2.3 Directed Acyclic Graphical Models (DAGM)

A directed acyclic graphical model over N random variables look like

$$p\left(x_{1:N}\right) = \prod_{i}^{N} p\left(x_i | x_{\pi_i}\right)$$

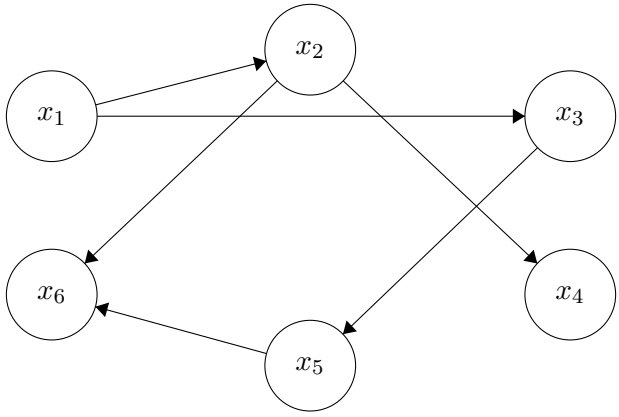
where x_i is a random variable and x_{π_i} denotes the parents of the node (which could be an empty set). This notion is more general than the fully dependence model that looked at above. Notice that here each node is only dependent on its parents rather than all other nodes. Thus, the complexity of such model reduces to exponential in fan-in of each node, instead of the total N.

2.3.1 Independence Assumption on DAGMs

Then, we have the independence relationship of[5] $x_i \perp x_{Ancestor(\pi_i)} | x_{\pi_i}$ which expands into

$$p\left(x_{1,\ldots,6}\right) = p\left(x_1\right) p\left(x_2|x_1\right) p\left(x_3|x_1\right) p\left(x_4|x_2\right) p\left(x_5|x_3\right) p\left(x_6|x_2, x_5\right)$$

with the following respective graph



[5]Requires topological ordering, to be added later.

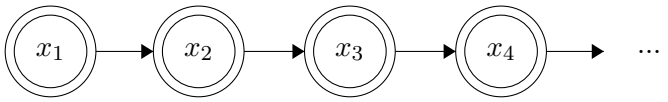
As we can see, the introduction of the assumption greatly reduced the complexity of the model.

2.3.2 (Example) Markov Chain

The following example has independence relationships that satisfies the Markov Property.

$$p(x) = p(x_1)p(x_2|x_1)p(x_3|x_2)p(x_4|x_3)...$$

and could be represented, in graphical model, as



2.4 Directed - Separation

2.4.1 (Definition) D - Separation

Directed-separation is a notion of connectedness in DAGs in which two (sets of) variables may or may not be connected conditioned on a third (set of) variable(s). D-connection implies conditional dependence and d-separation implies conditional independence.

2.4.2 Float Rules Derivation

Notation The double circles (or shaded circles) represent the notion of "conditioned-on".

Chain

In this case, we are interested in knowing whether or not

$$(X \perp Z)|Y$$

From the arrows between the nodes, we know that

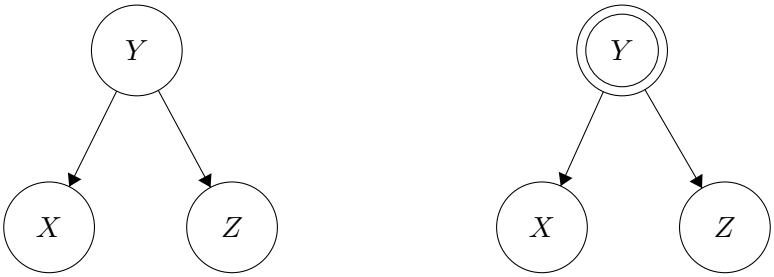
$$P(X, Y, Z) = P(X)P(Y|X)P(Z|Y)$$

Then, we have

$$\begin{aligned} P(X, Z|Y) &= \frac{P(X, Y, Z)}{P(Y)} \\ &= \frac{P(X)P(Y|X)P(Z|Y)}{P(Y)} \\ &= \frac{P(X, Y)P(Z|Y)}{P(Y)} \\ &= P(X|Y)P(Z|Y) \end{aligned}$$

and this completes the proof. ■

Common Clause



As previous, we are interested in knowing whether or not $(X \perp Z)|Y$. From the graph, we can know that

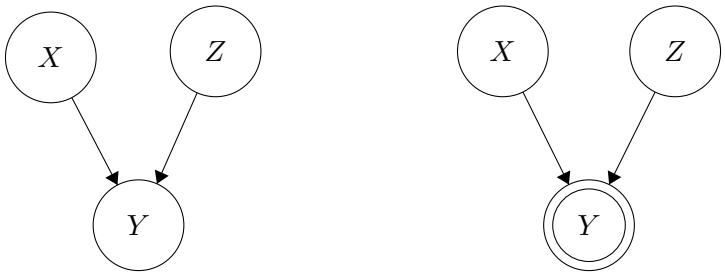
$$P(X, Y, Z) = P(Y)P(X|Y)P(Z|Y)$$

Then, we have

$$\begin{aligned} P(X, Z|Y) &= \frac{P(X, Y, Z)}{P(Y)} \\ &= \frac{P(Y)P(X|Y)P(Z|Y)}{P(Y)} \\ &= P(X|Y)P(Z|Y) \end{aligned}$$

and this completes the proof. ■

Explaining Away (Berkson's Paradox)



Notice from the graph that we have $P(X, Y, Z) = P(X)P(Z)P(Y|X, Z)$. First, I will prove that, in this case, $(X \not\perp Z)|Y$.

$$\begin{aligned} P(Z|X, Y) &= \frac{P(X)P(Z)P(Y|X, Z)}{P(X)P(Y|X)} \\ &= \frac{P(Z)P(Y|X, Z)}{P(Y|X)} \\ &\neq P(Z|Y) \end{aligned}$$

For marginal independence, i.e. $(X|Z)$, we want to show that $P(X,Z) = P(X)P(Z)$.

$$\begin{aligned} P(X,Z) &= \sum_{Y'} P(X,Y',Z) \\ &= \sum_{Y'} P(X)P(Z)P(Y'|X,Z) \\ &= P(X)P(Z)\sum_{Y'} P(Y'|X,Z) \\ &= P(X)P(Z) \end{aligned}$$

■

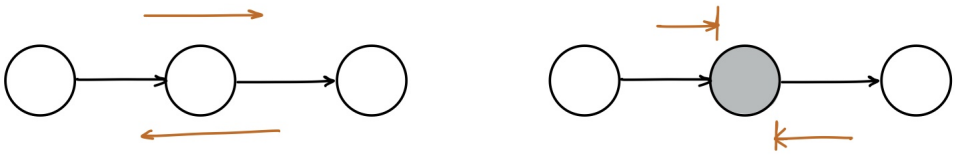
2.4.3 The Bayes-Ball Algorithm

To check if $x_A \perp x_B|x_C$, we will need

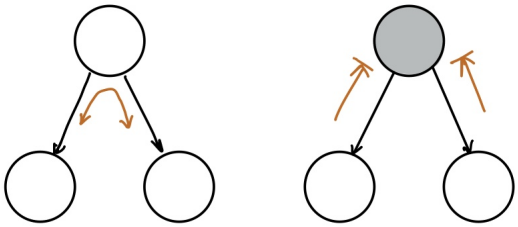
- Shade all nodes that were conditioned on, i.e. all nodes of x_C
- Place balls at each node in x_A (or x_B)
- Let the balls bounce around according to rules that are yet to be states below
  - If any of the balls reach any of the nodes in x_B from x_A (or reach x_A from x_B) then we declare $x_A \not\perp x_B|x_C$
  - Otherwise, $x_A \perp x_B|x_C$

2.4.4 Bayes Ball Rules

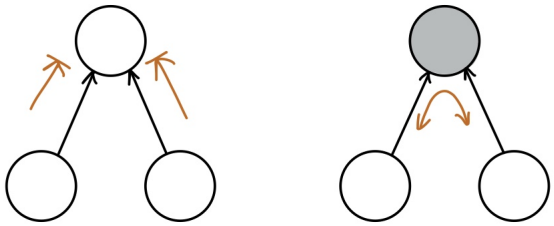
Chain



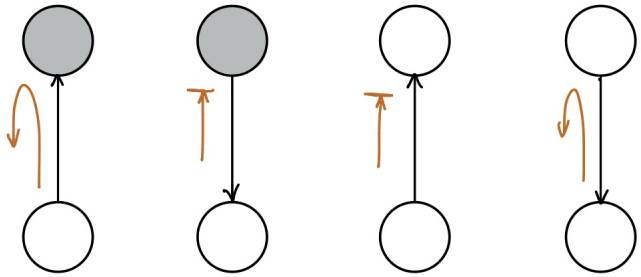
Common Cause



Explain Away



Linear



2.5 Unobserved Variables

Certain variables in our models may be unobserved , either some of the time or always, at training time or at test time. Graphically, we use shading to indicate observation.

2.5.1 Partially Unobserved Variables

[6] If variables are occasionally unobserved hen they are missing data, e.g., undefined inputs, missing class labels, erroneous target values. In this case, we can still model the joint distribution, but we marginalize the missing values

$$\begin{aligned}\ell(\theta;\mathcal{D}) &= \sum_{\text{complete}} \log p\left(x^c, y^c|\theta\right) + \sum_{\text{missing}} \log p\left(x^m|\theta\right) \\ &= \sum_{\text{complete}} \log p\left(x^c, y^c|\theta\right) + \sum_{\text{missing}} \log \sum_{y} p\left(x^m, y|\theta\right)\end{aligned}$$

2.5.2 Latent Variables

Above we discussed the case where some data are non-deterministically unobserved. Latent variables refers to those that ***never observed***. The handling of the latent variables depends on where it appears in our model,

- If we never condition on it when computing the probability of the variables we do observe, then we can just forget about it and integrate it out. For example, given y, x we fit the model

$$p(z, y|x) = p(z|y)p(y|x, w)p(w)$$

[6] An concrete example would be hospital data, where typically a large proportion of the data is missing.

- If z is not a leaf node, marginalizing over it will induce dependencies between its children. For example, given y, x we can fit the model

$$p(y|x) = \sum_z p(y|x,z)p(z)$$

2.5.3 Mixture Models

In this case, we are looking at data that has no input on class information. We can sum the labels out,

$$p(x|\theta) = \sum_{k=1}^{K} p\left(z=k|\theta_z\right) p\left(x|z=k,\theta_k\right)$$

where bayes rule comes in handy for calculating the posterior (class responsibilities) of the mixture component given some data, i.e.,

$$p\left(z=k|x,\theta_z\right) = \frac{p\left(z=k|\theta_z\right) p_k\left(x|\theta_k\right)}{\sum_j p\left(z=j|\theta_z\right) p_j\left(x|\theta_j\right)}$$

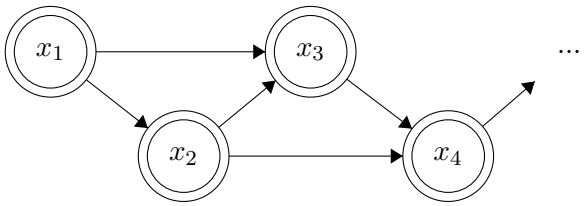
2.6 Examples

2.6.1 Second-order Markov Chain

Consider the model

$$p\left(\mathbf{x}_{1:T}\right) = p\left(x_1,x_2\right) p\left(x_3|x_1,x_2\right) p\left(x_4|x_2,x_3\right) \cdots = p\left(x_1,x_2\right) \prod_{t=3}^{T} p\left(x_t|x_{t-1},x_{t-2}\right)$$

which has the graphical representation (double circles mean "observed" here)

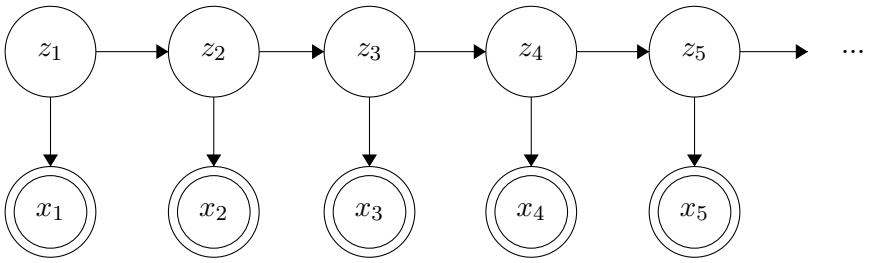


we notice that this model essentially assumes "the present depends on the past only through the current state as well as the last one."

2.6.2 Hidden Markov Models (HMMs)

HMM is a statistical model in which a system being modelled is assumed to be a Markov Process[7] with un-observed states. Here is the graphical model, where double circles mean "observed" here:

[7] In continuous time, the Markov Chain model is known as Markov Process



where

- z_t are hidden states taking on one of K discrete values
- x_t are observed variables taking on values in any space.

The above graph factorizes into

$$p\left(X_{1:T}, Z_{1:T}\right) = p\left(Z_{1:T}\right) p\left(X_{1:T} | Z_{1:T}\right) = p\left(Z_1\right) \prod_{t=2}^{T} p\left(Z_t | Z_{t-1}\right) \prod_{t=1}^{T} p\left(X_t | Z_t\right)$$

3 Lecture 4 - Exact Inference

3.1 Variable Elimination

3.1.1 (Simple Example) Chain

The example that we will consider is the simple chain

$$A \to B \to C \to D$$

where we want to compute $P(D)$, with no observation for other variables. We have

$$X_F = \{D\}, X_E = \{\}, X_R = \{A, B, C\}$$

No observations for other variables means that we have no 'evidence' for other variables.

X_F is the set of variable that we are interested in, X_E is evidence, and X_R is the set of extraneous variables, i.e. what we marginalize out

The graphical model gives the factorization of

$$p(A, B, C, D) = p(A)p(B|A)p(C|B)p(D|C)$$

thus, if we want to find $p(D)$ we can marginalize over all other variables, i.e.

$$\begin{aligned} p(D) &= \sum_{A,B,C} p(A, B, C, D) \\ &= \sum_{C} \sum_{B} \sum_{A} p(A)p(B|A)p(C|B)p(D|C) \end{aligned}$$

If we marginalize the above the naïve way, then it cost exponentially $\mathcal{O}(k^n)$. Thus, we need to find a ***elimination ordering*** that gives us a smaller runtime complexity. We can reduce the complexity by first computing terms that appear across the other marginalization sums,

Here ϕ means factor, an non-normalized "pmf"

$$\begin{aligned} \phi(D) &= \sum_{C} p(D|C) \sum_{B} p(C|B) \sum_{A} p(A)p(B|A) \\ &= \sum_{C} p(D|C) \sum_{B} p(C|B)\phi(B) \\ &= \sum_{C} p(D|C)\phi(C) \end{aligned}$$

3.2 Sum-Product Inference

Let $X = Z \cup Y \wedge Z \cap Y = \emptyset$ be a set of random variables. . Then, to compute $P(Y)$ for a directed and undirected model could be achieved by the sum product inference algorithm

$$\tau(Y) = \sum_{z} \prod_{\phi \in \Phi} \phi\left(\text{Scope}(\phi) \cap Z, \text{Scope}(\phi) \cap Y\right) \quad \forall Y$$

where the scope for each ϕ simply means the set of all random variables that have appeared in that specific ϕ, and Φ denotes the set of all ϕ's.

Here Y is the set that we want to do inference on and thus we are marginalizing over everything in Z

how to generalize to undirected graph?

3.2.1 SP - Inference in Directed Models

In a directed model, the Φ is given by the conditional probability distributions for all variables, i.e.

$$\Phi = \{\phi_{x_i}\}_{i=1}^{N} = \{p\left(x_i | \text{ parents } (x_i)\right)\}_{i=1}^{N}$$

The resulting term $\tau(Y)$ will be automatically normalized.

3.2.2 SP - Inference in Undirected Models

For undirected models, Φ is given by the set of un-normalized potentials, and we ***must*** normalize the resulting $\tau(Y)$ by $\sum_Y \tau(y)$

3.2.3 (Example) Directed Graph

We have the following factorization of the joint distribution

$$p(C, D, I, G, S, L, H, J) = p(C)p(D|C)p(I)p(G|D, I)p(L|G)P(S|I)p(J|S, L)p(H|J, G)$$

and we will need the following potentials

$$\Phi = \{\phi(C), \phi(C, D), \phi(I), \phi(G, D, I), \phi(L, G), \phi(S, I), \phi(J, S, L), \phi(H, J, G)\}$$

consider the problem where we want to infer $P(J)$ with elimination ordering $\prec_{\{C,D,I,H,G,S,L\}}$.[8] Then,

$$
\begin{aligned}
p(J) &= \sum_L \sum_S \phi(J,L,S) \sum_G \phi(L,G) \sum_H \phi(H,G,J) \sum_I \phi(S,I)\phi(I) \sum_D \phi(G,D,I) \sum_C \phi(C)\phi(C,D) \\
&= \ldots \underbrace{\sum_C \phi(C)\phi(C,D)}_{\tau(D)} \\
&= \ldots \underbrace{\sum_D \phi(G,D,I)\tau(D)}_{\tau(G,I)} \\
&= \ldots \underbrace{\sum_I \phi(S,I)\phi(I)\tau(G,I)}_{\tau(S,G)} \\
&= \ldots \underbrace{\tau(S,G) \sum_H \phi(H,G,J)}_{\tau(S,G)\tau(G,J)} \quad \text{// Note that } \tau(S,G) \text{ doesn't contain } H \\
&= \ldots \underbrace{\sum_G \phi(L,G)\tau(S,G)\tau(G,J)}_{\tau(J,L,S)} \\
&= \ldots \underbrace{\sum_S \phi(J,L,S)\tau(J,L,S)}_{\tau(J,L)} \\
&= \ldots \underbrace{\sum_L \tau(J,L)}_{\tau(J)} \\
&= \tau(J)
\end{aligned}
$$

3.2.4 Complexity of VE

The complexity of VE is

$$\mathcal{O}\left(mk^{N_{\max}}\right)$$

where

- $m = |\Phi|$ is the number of initial factors, i.e. the number of terms in the original factorization of the joint probability (given the graphical model)
- k is the number of states each random variable takes (assumed to be equal here)
- N_i is the number of random variables inside each sum $\sum_i$ *at the time of marginalization*
- $N_{max} \triangleq \max_i N_i$ is the number of random variables inside the largest sum. (Inner sums are counted as one term all together)

[8] Note that what comes first in the ordering is the what we want to sum over in the inner most summation

4 Lecture 5 - Message passing, Hidden Markov Models, and Sampling

4.1 Message Passing

4.1.1 Belief Propagation: Motivation and Definitions

Our goal is to compute the marginal of every variable in graph $p(x_i), \forall x_i \in X$. Notice that in a tree, we have

$$P\left(X_{1:n}\right) = \frac{1}{Z} \prod_{k \in \{1,\ldots,n\}} \phi\left(x_k\right) \prod_{(i,j) \in T} \phi_{i,j}\left(x_i, x_j\right)$$

Below, T means the set of edges in the tree

and thus, if we want to compute $P(X_1)$ we can marginalize out all other variables

$$\begin{aligned} P(X_1) = \sum_{x_2,\ldots,x_n} P(X_{1:n}) &= \sum_{x_2,\ldots,x_n} \frac{1}{Z} \prod_k \phi\left(x_k\right) \prod_{(i,j) \in T} \phi_{i,j}\left(x_i, x_j\right) \\ &\propto \sum_{x_2,\ldots,x_n} \prod_k \phi\left(x_k\right) \prod_{(i,j) \in T} \phi_{i,j}\left(x_i, x_j\right) \end{aligned}$$

Now that we have the goal, we can group what we want to compute into a more structured representation, where we define ***message-passing*** from variable j to $i \in N(j)$ as

$$m_{j \to i}\left(x_i\right) = \sum_{x_j} \phi_j\left(x_j\right) \phi_{ij}\left(x_i, x_j\right) \prod_{k \in N(j) \neq i} m_{k \to j}\left(x_j\right)$$

4.1.2 Belief Propagation Algorithm

1. Choose an root r arbitrarily,
2. Pass messages from leaves to r
3. Pass messages from r to leaves
4. Compute

$$p\left(x_i\right) \propto \phi_i\left(x_i\right) \prod_{j \in \mathcal{N}(i)} m_{j \to i}\left(x_i\right), \forall_i$$

Step 2 and 3 here so that we will compute all the $m_{i \to j}, \forall (i,j), (j,i) \in T$, after which we can use them to calculate the marginals using formula in step 4

Notice that the idea is that we want to calculate ***all*** of the marginals

4.2 Inference in Hidden Markov Models

There are the following four kinds of inference that we can do on a HMM, namely

- ***Filtering:*** Compute the belief state $p\left(z_t | \mathbf{x}_{1:t}\right)$ online. Notice that we call this inference operation filtering because this produces a result smoother than simply computing $p(z_t|\mathbf{x}_t)$.
- ***Smoothing:*** Compute $p\left(z_t | \mathbf{x}_{1:T}\right)$ offline. Notice that this computation happens after all $\mathbf{x}_{1:T}$ observations have been collected and hence is offline.[9]

Definition in KPM is different from course note

[9] A very good intuitive example from KPM is that we have a detective investigating a crime scene, and as we gather more observation, the prediction that we make tends to be more and more certain and less 'stochastic'.

- ***Fixed lag smoothing:*** Compute $p\left(z_{t-\ell}|\mathbf{x}_{1:t}\right)$, where $\ell > 0$ is called the lag. Essentially, we want to do inference on the hidden state z_α and we wait until we observe all of $\mathbf{x}_{1:\alpha+\ell}$ before we carry out the computation.

- ***Prediction:*** In this case, we want to compute $p\left(z_{t+h}|\mathbf{x}_{1:t}\right)$ for some $h > 0$. (h is called the prediction horizon) Let's first take a look at the example where $h = 2$, then

$$\begin{aligned} p(z_{t+2}|\mathbf{x}_{1:t}) &= \sum_{z_{t+1}} \sum_{z_t} p(z_t, z_{t+1}, z_{t+2}|\mathbf{x}_{1:t}) \\ &= \sum_{z_{t+1}} \sum_{z_t} p(z_t|\mathbf{x}_{1:t}) p(z_{t+1}|z_t, \mathbf{x}_{1:t}) p(z_{t+2}|z_{t+1}, z_t, \mathbf{x}_{1:t}) \\ &= \sum_{z_{t+1}} \sum_{z_t} p\left(z_{t+2}|z_{t+1}\right) p\left(z_{t+1}|z_t\right) p\left(z_t|\mathbf{x}_{1:t}\right) \end{aligned}$$

  where the last step of simplification could be easily justified using Bayes Ball rules. Above, we computed the prediction about the future hidden states, and it can be converted into prediction about the future observations using

$$p\left(\mathbf{x}_{t+h}|\mathbf{x}_{1:t}\right) = \sum_{z_{t+h}} p\left(\mathbf{x}_{t+h}|z_{t+h}\right) p\left(z_{t+h}|\mathbf{x}_{1:t}\right)$$

  which is called ***Posterior Predictive Density***.

4.3 The Forward - Backward Algorithm

Suppose that we want to find $p(z_t|x_{1:T}), \forall t$, then it can be broken down into two pieces

- ***Forward Filtering:*** Compute $p\left(z_t, x_{1:t}\right) \quad \forall t$, which effectively is the probability of the hidden state z_t given all past observations, up to including x_t.

- ***Backward Filtering:*** Compute $p\left(x_{1+t:T}|z_t\right) \quad \forall t$, which computes the probability of *all* future observations from x_{t+1} up to including x_T given the current hidden state, i.e. z_t.

Then, the complete smoothing $p(z_t|x_{1:T})$ could be computed as

This is called **Latent sequence given observations in HMM** in the course notes

$$\begin{aligned} p\left(z_t|x_{1:T}\right) &= p\left(x_{1:T}\right) p\left(z_t, x_{1:T}\right) \\ &\propto p\left(z_t, x_{1:T}\right) \\ &= p\left(z_t, x_{1:t}\right) p\left(x_{t+1:T}|z_t, x_{1:t}\right) \\ &= p\left(z_t, x_{1:t}\right) p\left(x_{t+1:T}|z_t\right) && \text{// By Bayes Ball} \\ &= \text{(Forward Recursion)(Backward Recursion)} \end{aligned}$$

4.3.1 Forward Filtering Recursion

$$\begin{aligned}\alpha_t(z_t) = p\left(z_t, x_{1:t}\right) &= \sum_{z_{t-1}} p\left(z_{t-1}, z_t, x_{1:t}\right) \\ &= \sum_{z_{t-1}} p\left(x_t | z_{t-1}, z_t, x_{1:t-1}\right) p\left(z_t | z_{t-1}, x_{1:t-1}\right) \underbrace{p\left(z_{t-1}, x_{1:t-1}\right)}_{=\alpha_{t-1}(z_{t-1})} \\ \implies \alpha_t\left(z_t\right) &= p\left(x_t | z_t\right) \sum_{z_{t-1}} p\left(z_t | z_{t-1}\right) \alpha_{t-1}\left(z_{t-1}\right)\end{aligned}$$

Notice that our forward recursion contains our emission, $p(x_t|z_t)$ and transition $p(z_t|z_{t-1})$. The base case of the recursion could be unwinded into

$$\alpha_1\left(z_1\right) = p\left(z_1, x_1\right) = p\left(z_1\right) p\left(x_1 | z_1\right)$$

4.3.2 Backward Filtering Recursion

$$\begin{aligned}p\left(x_{t+1:T} | z_t\right) &= \sum_{z_{t+1}} p\left(z_{t+1}, x_{t+1:T} | z_t\right) \\ &= \sum_{z_{t+1}} p\left(x_{t+2:T} | z_{t+1}, z_t, x_{t+1}\right) p\left(x_{t+1} | z_{t+1}, z_t\right) p\left(z_{t+1} | z_t\right) \\ \implies \beta_t\left(z_t\right) &= \sum_{z_{t+1}} \underbrace{p\left(x_{t+2:T} | z_{t+1}\right)}_{=\beta_{t+1}(z_{t+1})} p\left(x_{t+1} | z_{t+1}\right) p\left(z_{t+1} | z_t\right)\end{aligned}$$

and hence if we recurse the above relationship, we will unwind to the base case

$$\beta_1\left(z_1\right) = p\left(x_{3:T} | z_2\right) p\left(x_2 | z_2\right) p\left(z_2 | z_1\right)$$

4.4 Sampling

The word sample here refers to a single realization from a distribution rather than a set.

The goal is to,

- Generate sample $\left\{x^{(r)}\right\}_{r=1}^{R}$ from a probability distribution $p(x)$, and/or
- Estimate expectations of functions $f(x)$ under some distribution $p(x)$, usually we want to estimate moments for a distribution

$$E = \mathop{\mathbb{E}}_{x \sim p(x)}[f(x)] = \int f(x) p(x) dx$$

4.4.1 Ancestral Sampling

Generating marginal samples If you are only interested in sampling a particular set of nodes, you can simply sample from all the nodes jointly, then ignore the nodes you don't need.

Generating conditional samples If you want to sample conditional on a node with no parents, that's also easy - you can simple do ancestral sampling starting from the nodes you have.

(#! Important:) However, to sample from a DAG conditional on leaf nodes is hard in the same way that inference is hard in general. E.g. sampling the unknown key in a crypto-system given the cypher-text but not knowing the plaintext. Finding ways to do this approximately is what a lot of the rest of the course will be about.

4.4.2 Simple Monte Carlo

(Definition) - Simple MC Given $\left\{x^{(r)}\right\}_{r=1}^{R} \sim p(x)$, we want to estimate the expectation $\mathop{\mathbb{E}}_{x\sim p(x)}[f(x)]$.

$$E = \mathop{\mathbb{E}}_{x\sim p(x)}[f(x)] \approx \frac{1}{R}\sum_{r=1}^{R} f\left(x^{(r)}\right) = \hat{E}$$

Unbiasedness of MC

$$\begin{aligned}
\mathbb{E}_{x\sim p\left(\left\{x^{(i)}\right\}_{r=1}^{R}\right)}[\hat{E}] &= \mathbb{E}\left[\frac{1}{R}\sum_{r=1}^{R} f\left(x^{(r)}\right)\right] \\
&= \frac{1}{R}\sum_{r=1}^{R}\mathbb{E}\left[f\left(x^{(r)}\right)\right] \\
&= \frac{1}{R}\sum_{r=1}^{R}\mathop{\mathbb{E}}_{x\sim p(x)}[f(x)] \\
&= \frac{R}{R}\mathop{\mathbb{E}}_{x\sim p(x)}[f(x)] \\
&= E
\end{aligned}$$

■

Variance of MC

$$\begin{aligned}
\operatorname{var}[\hat{E}] &= \operatorname{var}\left[\frac{1}{R}\sum_{r=1}^{R} f\left(x^{(r)}\right)\right] \\
&= \frac{1}{R^2}\operatorname{var}\left[\sum_{r=1}^{R} f\left(x^{(r)}\right)\right] \\
&= \frac{1}{R^2}\sum_{r=1}^{R}\operatorname{var}\left[f\left(x^{(r)}\right)\right] \\
&= \frac{1}{R}\operatorname{var}\left[f(x)\right]
\end{aligned}$$

We notice that the accuracy of MC estimates only depends on the variance of f, not on the dimension of x. Also, we the number of samples, R, increases, the variance $\hat{E}$ will decrease at a rate proportional to $\frac{1}{R}$. ■