Probabilistic Machine Learning

Seong-Hwan Jun

Logistics

BST 570-1 Probabilistic Machine Learning (4 credits)

- Time: Tuesday/Thursday 11AM - 12:40PM.
- Location: SRB 1.416.
- Office hour: Wednesday 12-1PM. SRB BST Yakovlev room.

Course objectives

- Learn to formulate custom probabilistic models for real world problems.
- Understand the principles of probabilistic reasoning.
- Gain familiarity with inference and learning methods.
- Gain familiarity with machine learning software libraries and add Python to your research toolbox.

Textbooks

Books written by Kevin Murphy.

- Probabilistic Machine Learning (PML1 and PML2) – draft PDFs are freely available for download.
- Machine Learning: a Probabilistic Perspective (MLPP). This is an older version but well-written and combines most of the contents from PML1 and PML2 into one book.

The majority of the topics covered in this course are from PML2. The first module will primarily use PML1.

Outline

- Module 1: Foundation
- Module 2: Exact Inference
- Module 3: Exactly Approximate Inference
- Module 4: Approximately Approximate Inference
- Module 5: Topics and trends in probabilistic machine learning

Assessment

- 15% x 4 assignments.
  - Due 3 weeks from the release date, except the first one is due in 2 weeks from release.
- 10% literature review:
  - Read up on a topic and give a 40 minute presentation.
  - Critically read and assess machine learning research.
- 30% final project.
  - Develop an end-to-end probabilistic machine learning method for a problem of your choice.

Schedule

- Jan 28: A1 release.
- Feb 11: A1 due/A2 release.
- Mar 4: A2 due/A3 release.
- Mar 10-14: Spring break.
- Mar 27: A3 due/A4 release.
- Apr 17: A4 due.
- Apr 22-May 1: Presentations.
- May 8: Project due.

Teaching style

We have 100 minutes for each class.

- 50 minute lecture.
- 10 minute break.
- 40 minutes: practicum/demo/lecture.

- We will write or look at some code together.
- Or just a plain lecture if we need more time to cover the materials.

Module 1: Foundation

- What is probabilistic machine learning? (PML1 Ch. 1).
- Probability as an extension of logic (PML1 Ch. 2 and 4).
- Probabilistic graphical models (PML1 Ch. 3.6 and PML2 Ch. 4).
- Optimization and sampling (PML1 Ch. 8 and PML2 Ch. 6).
- Python/R libraries for ML.

What is machine learning?

According to Tom Mitchell (CMU professor):

"A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured in P, improves with experience E."

- E, experience, is essentially data. What are the common data types and structures?
- Examples of P, T?

Supervised learning

- $E = \{(x_i, y_i)\}_{i=1}^N$ where $x_i \in \mathcal{X}$ and $y_i \in \mathcal{Y}$.
- $\mathcal{X}$ denotes the input space, usually represented in a tabular form ($N \times D$ matrix – e.g., design matrix).
- $\mathcal{Y}$ denotes the output space.
- The goal is to "learn" a mapping:

$$f : \mathcal{X} \to \mathcal{Y}. \tag{1}$$

Here, "learn" means to find f that optimizes performance measure P. How?

Example: regression

If we have $\mathcal{Y} = \mathbb{R}^d$ and we choose quadratic loss as our performance measure:

$$P(E) = \sum_i (f(x_i) - y_i)^2 \tag{2}$$

What does it mean to optimize P?

Example: regression

Let us further assume f is a linear function of $x \in \mathcal{X}$:

$$f_\beta(x) = \beta^T x, \tag{3}$$

then we are optimizing the parameters β of the linear model. Essentially, learning in this context amounts to solving a linear regression problem.

Example: classification

- Let $\mathcal{Y}$ be a discrete set.
- If $\mathcal{Y} = \{0, 1\}$, we have a binary classification problem.
- If $\mathcal{Y} = \{1, ..., K\}$, we have a multi-class classification problem.
- What are some of the choices for f and P?

Example: binary classification

If $\mathcal{Y} = \{0, 1\}$ and we assume that $y_i \sim \text{Bernoulli}(\theta)$, then we may choose the following model:

$$\theta = f_\beta(x) = \frac{\exp(\beta^T x)}{1 + \exp(\beta^T x)}, \tag{4}$$

and

$$P(E) = \prod_i \theta_i^{y_i} (1 - \theta_i)^{1 - y_i}. \tag{5}$$

In this case, P is a likelihood and we are essentially fitting a logistic regression model.

Unsupervised learning.

- In many cases the label may be hard to obtain or unavailable – so $E = \{x_i\}_{i=1}^N$.
- We can think of the label as being hidden (latent) and infer its value: **clustering**.
- If x is high-dimensional, we may wish to learn the low dimensional embedding z that underlies x: **dimension reduction**.

Example: Gaussian mixture model

$$x_i|z_i \sim \text{Normal}(\mu_{z_i}, \Sigma_{z_i}) \tag{6}$$

$$z_i \sim \text{Categorical}(\pi). \tag{7}$$

The performance measure P may be given by complete log-likelihood:

$$P = \sum_{i=1}^{N} \log \sum_{z=1}^{K} p(x_i|\mu_z, \Sigma_z)p(z). \tag{8}$$

Example: dimension reduction

$$X = ZW. \tag{9}$$

- X has dimension $N \times D$.
- Z has dimension $N \times K$.
- W has dimension $K \times D$.
- $K \ll D$.
- Different performance measure/optimization objective leads to a different solution: e.g., PCA, non-negative matrix factorization, factor analysis, etc.

Example: autoencoders

- Encoder: $f_\beta : \mathcal{X} \to \mathcal{Z}$.
- Decoder: $g_\theta : \mathcal{Z} \to \mathcal{X}$.
- The dimension of $\mathcal{Z}$ is smaller than the dimension of $\mathcal{X}$; $z \in \mathcal{Z}$ is referred to as latent feature representation of x.
- Minimize reconstruction error, e.g., mean squared error:

$$P(X, g(f(X)) = \frac{1}{N} \sum_i (x_i - g_\theta(f_\beta(x_i)))^2. \tag{10}$$

Generative models

- Many problems involve unstructured data and mixed data types.
- A **generative model** is a joint probability distribution $p(x)$, with some observed and some latent/hidden.
- To specify the generative model, it is sufficient to specify the conditional distributions.
- **"Learning the generative model" is the main theme of this course**.

Example: Gaussian mixture model

We observe $y_i \in \mathbb{R}$. Let $z_i \in \{1, ..., K\}$ denote the latent clustering membership. We can specify a generative model as follows:

- $p_\pi(z_i)$: Categorical distribution with parameters π.
- $p_\theta(y_i|z_i)$: Gaussian distribution with $\theta = (\mu_{z_i}, \sigma^2_{z_i})$.

The subscript is used to denote model parameters, i.e., θ, π. These are variables that are needed to specify the generative model but may not be primary variables of interest.

Generally, we are content with point estimates for these parameters.

Example: Gaussian mixture model

Let $z_{1:N} = (z_1, ..., z_N)$ and $y_{1:N} = (y_1, ..., y_N)$.

The joint probability distribution over the *primary variables* is given by,

$$p_{\theta,\pi}(y_{1:N}, z_{1:N}) = p_\pi(z_{1:N}) p_\theta(y_{1:N}|z_{1:N}) \tag{11}$$

$$= \prod_i p_\pi(z_i) p_\theta(y_i|z_i), \tag{12}$$

where we use independence assumption of the data to decompose the joint pdfs for $z_{1:N}, y_{1:N}$.

Example: Gaussian mixture model

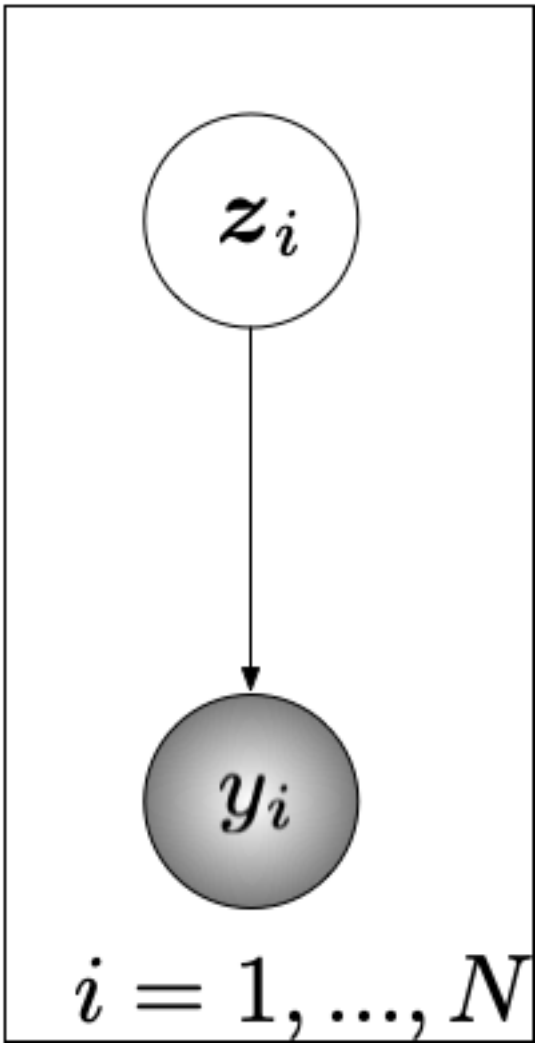


Figure 1: Graphical model for GMM. The nodes represent random variables. Shaded nodes indicate observed variables. Plates are useful for representing repetition (rectangle). Parent-child relationship captures conditional dependence of child y_i on parent z_i.

Example: Gaussian mixture model

How do we infer z_i?

What does it mean to infer it?

How do we learn θ?

The distinction becomes a bit blur at times, especially when you operate in Bayesian framework.

Example: Gaussian mixture model

Inference: we are interested in the distribution over z_i.

By Bayes theorem:

$$p(z_{1:N}|y_{1:N}) = \frac{p(y_{1:N}|z_{1:N})p(z_{1:N})}{p(y_{1:N})} \tag{13}$$

$$= \frac{p(y_{1:N}|z_{1:N})p(z_{1:N})}{\sum_{z_{1:N}} p(y_{1:N}, z_{1:N})}. \tag{14}$$

This is the posterior distribution over $z_{1:N}$ after having observed $y_{1:N}$.

Example: Gaussian mixture model

Independence assumption over n:

$$p(z_{1:N}|y_{1:N}) = \frac{\prod_n p(y_n|z_n)p(z_n)}{p(y_{1:N})}. \tag{15}$$

Example: Gaussian mixture model

Marginal likelihood:

$$p(y_{1:N}) = \sum_{z_{1:N}} p(y_{1:N}|z_{1:N})p(z_{1:N}) \tag{16}$$

$$= \sum_{z_{1:N}} \prod_i p(y_i|z_i)p(z_i) \tag{17}$$

$$= \prod_i \sum_{z_i} p(y_i|z_i)p(z_i) \tag{18}$$

$$= \prod_i p(y_i). \tag{19}$$

Example: Gaussian mixture model

Can we update our belief over z_i after having observed $y_{1:N}$?

$$p(z_i|y_{1:N}) = \sum_{z_{-i}} p(z_{1:N}|y_{1:N}) \tag{20}$$

$$= \frac{\sum_{z_{-i}} \prod_n p(y_n|z_n)p(z_n)}{p(y_{1:N})} \tag{21}$$

$$= \frac{p(y_i|z_i)p(z_i)\prod_{n\neq i}\sum_{z_n} p(y_n|z_n)p(z_n)}{p(y_{1:N})} \tag{22}$$

$$= \frac{p(y_i|z_i)p(z_i)}{p(y_i)}. \tag{23}$$

Example: Gaussian mixture model

We implicitly assumed π was known in the above derivation.

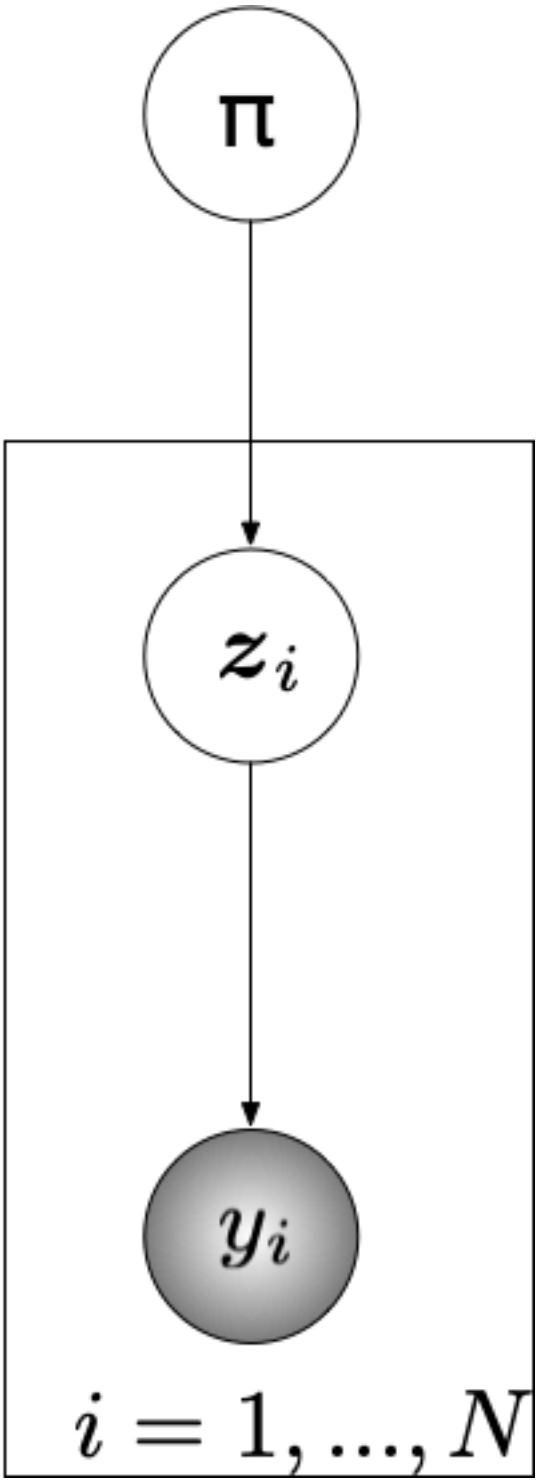
Let's represent π as a primary variable of interest.

$$\pi \sim \text{Dirichlet}(\alpha) \tag{24}$$

$$z_i \sim \text{Categorical}(\pi) \tag{25}$$

$$y_i|z_i \sim \text{Normal}(\mu_{z_i}, \Sigma_{z_i}) \tag{26}$$

Example: Gaussian mixture model

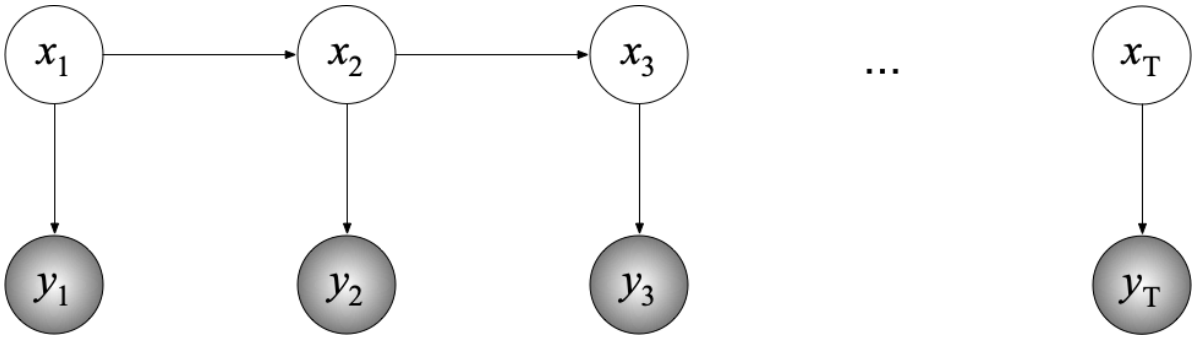


Exercises: how does this change the derivation for $p(z_i|y_{1:N})$? How do we infer π and learn θ? We will get to this in Module 2.

Graphical models

- Modeling language for probabilistic machine learning.
- Provides a framework to specify a generative model and communicate dependence structure between the variables and hence, independence relationships.
- Depending on the structure the graphical model, there are known efficient inference algorithms.
- Some common examples will be covered in-depth in the course. But here comes the preview.

Example: hidden Markov model



$$X_1 \sim \mu(\cdot) \tag{27}$$
$$X_t|X_{t-1} \sim f(\cdot|X_{t-1}) \tag{28}$$
$$Y_t|X_t \sim g(\cdot|X_t). \tag{29}$$

Example: hidden Markov model

- $X_{1:T}$: hidden states.
- X_1 is sampled from some initial distribution μ.
- Subsequent X_t depends only on the previous state (first-order Markov chain). Requires specification of the *transition kernel* f.
- Observation depends on x_t, e.g., it may be some noisy version of x_t. Requires specification of the *emission model* g.

Example: hidden Markov model

AR(1) or autoregressive model of order 1 can be posed as an example of HMM.

AR(1) is given by the transition dynamic: $X_t = \phi X_{t-1} + \epsilon_t$. We can re-write this as,

$$X_t = \phi X_{t-1} \tag{30}$$
$$Y_t = X_t + \epsilon_t. \tag{31}$$

And we only observe the noisy version Y_t and want to infer the latent X_t.

Example: hidden Markov model

Robot localization (Roomba):

- X_t represents 2-D coordinate at time t.
- The system observes Y_t from sensors, e.g., $Y_t = X_t + \epsilon$ for some mean zero (Gaussian) noise ϵ.
- $X_t|X_{t-1}$: determined by transition dynamic f based on the velocity, acceleration, terrain (friction), obstacles, and possibly noise δ.
- Bayesian filtering problem: system maintains probability distribution over X_t.

Tree structured graphical model

E.g., phylogenetic trees.

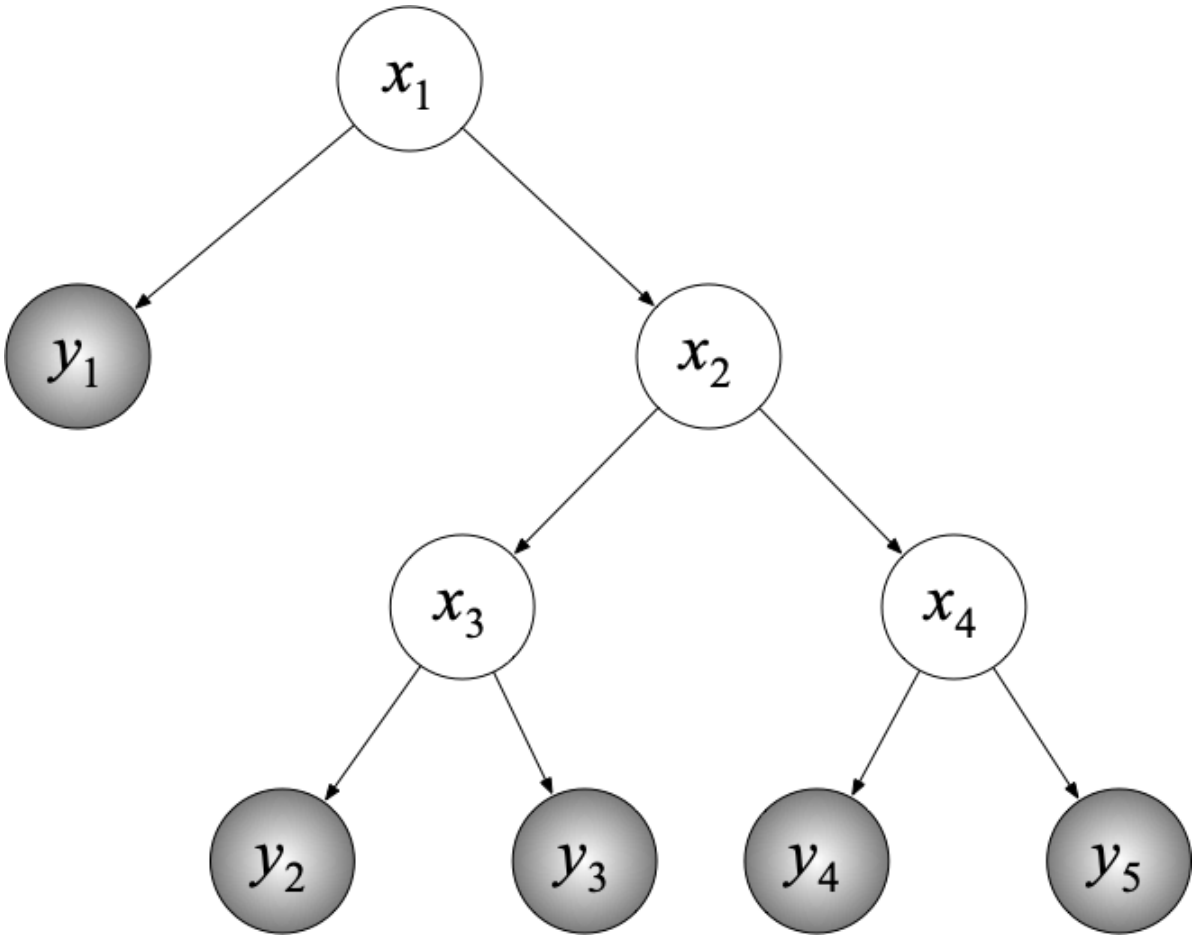


Figure 2: Observations at the leaves can be viral strands, species, cancer cells, etc depending on the problem.

Tree structured graphical model

Which traits are shared? Which species are most closely related?

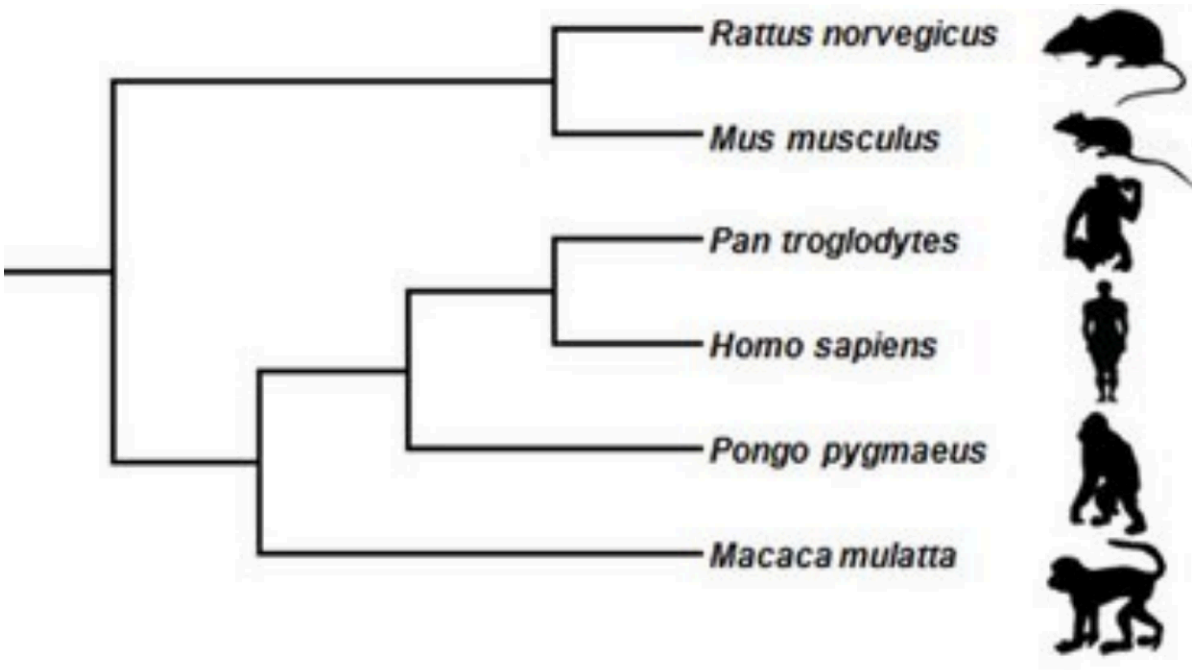


Figure 3: https://www.researchgate.net/figure/Phylogenetic-tree-of-the-species-used-for-the-evolutionary-analysis-of-Hox-genes_fig3_245029837

Tree structured graphical model

Nextstrain

Tree structured graphical model

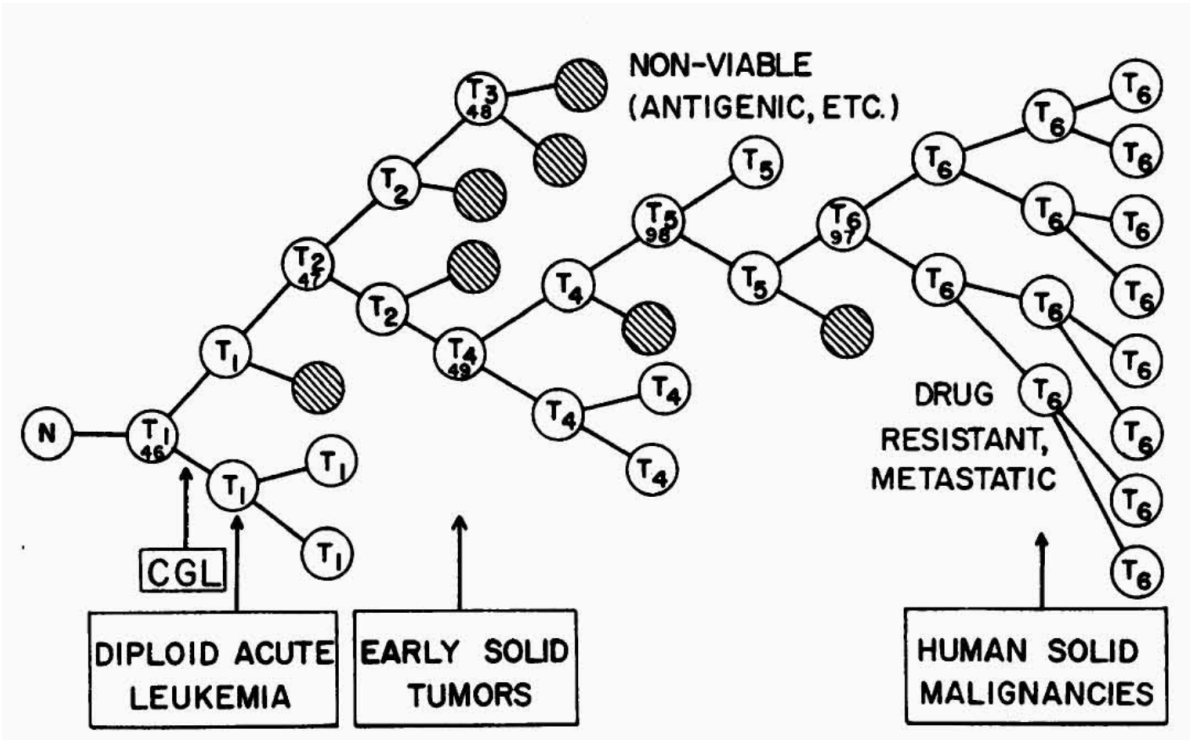


Figure 4: Cancer evolution (Nowell 1976)

Tree structured graphical model

- Tree structure is denoted $T = (V, E)$. V denotes nodes and E denotes edges. Trees are acyclic.
- Let $X = (X_v)$ denote a collection of random variables represented by the nodes of the tree. Joint distribution is given by,

$$p(X) = \prod_{v \in V} p(x_v | x_{pa(v)}), \tag{32}$$

where $pa(v)$ denotes the parents of v. For trees, $|pa(v)| = 1$ except for the root node r, where $pa(r) = \emptyset$.

Graphical models

- $O \subset V$ set of observed nodes
- $H \subset V$ set of latent (hidden) nodes.
- Depending on the context, the model parameters will be explicitly denoted by θ or as part of H, the hidden variables.

We wish to compute $P(X_H|X_O)$. How to do this in a principled manner?

Inference and Learning

What does it mean to "infer" and "learn"?

- Infer in this course generally means to compute the posterior distribution over the hidden variables.
- Learn is used for the (hyper) parameters: how to learn or estimate the parameters, e.g., point estimates.

Given observation y and hidden variable z and parameters θ. How do we estimate θ while eliminating or accounting for the contribution of uncertainty from z?

Why Bayesian?

- Natural mechanism for updating our beliefs over random variables.
- Being Bayesian usually means that the performance measure is going to be the posterior distribution.
- Computing the posterior captures the inherent uncertainty associated with the variable.

Probabilistic machine learning

- Relies on probability theory to specify relationship between random variables and model uncertainty.
- Probabilistic graphical models serve as a modeling language.
- Probabilistic reasoning via Bayesian statistics and computational techniques.
- Development of software libraries in the last decade has allowed for wide array of custom model specifiction and ease of inference (almost automatic!).

Summary

- We are interested in the generative model: $p(hidden, observed)$ or $p(data, label)$.
  - This is in contrast to discriminative models that aims to learn $p(label|data)$.
- Graphical model serves as a modeling language but also helps you to clarify dependence structure and design inference algorithm.
- Bayesian statistics makes it natural to update your belife given observation.
- We will study the history of probabilistic machine learning leading up to the generative pre-trained transformers (GPTs) or generative AI.

Notation

- Although we used P to refer to performance metric, we will use L for loss moving forward.
- p is reserved for probability distribution.
- P is reserved for probability measure.

Readings for Module I (Suggested not mandatory)

- Probability and Statistics (PML2 Ch 2-3).
- Graphical models (PML2 Ch 4).
- Monte Carlo sampling basics (PML2 Ch 11).
- Optimization (PML1 Ch 8).