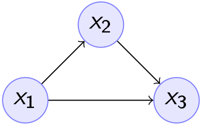
Generative Modelling

Christos Dimitrakakis

November 1, 2024

Outline

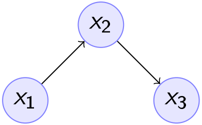
Graphical models



- Variables x_1, x_2, x_3
- Arrows denote dependencies between variables.

Conditional independence

Example



Graphical model for the factorisation $\mathbb{P}(x_3 \mid x_2)\,\mathbb{P}(x_2 \mid x_1)\,\mathbb{P}(x_1)$.

Definition

- Consider variables $x_1, \ldots, x_n$.
- Let B, D be subsets of $[n]$.

We say x_i is conditionally independent of $\boldsymbol{x}_B$ given $\boldsymbol{x}_D$ and write

$$x_i \perp\!\!\!\perp \boldsymbol{x}_B \mid \boldsymbol{x}_D$$

if and only if:

$$\mathbb{P}(x_i, \boldsymbol{x}_B \mid \boldsymbol{x}_D) = \mathbb{P}(x_i \mid \boldsymbol{x}_D)\,\mathbb{P}(\boldsymbol{x}_B \mid \boldsymbol{x}_D).$$

Directed graphical model

A collection of n random variables $x_i : \Omega \to X_i$, and let $X \triangleq \prod_i X_i$, with underlying probability measure P on Ω. Let $\boldsymbol{x} = (x_i)_{i=1}^n$ and for any subset $B \subset [n]$ let

$$\boldsymbol{x}_B \triangleq (x_i)_{i \in B} \tag{1}$$

$$\boldsymbol{x}_{-j} \triangleq (x_i)_{i \neq i} \tag{2}$$

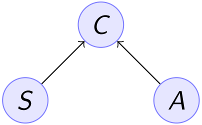
Model specification

$$x_1 \sim f \tag{3}$$

$$x_2 \mid x_1 = a \sim g(a) \tag{4}$$

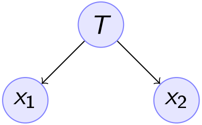
$$x_3 \mid x_2 = b \sim h(b), \tag{5}$$

Smoking and lung cancer



Smoking and lung cancer graphical model, where S: Smoking, C: cancer, A: asbestos exposure.

Time of arrival at work



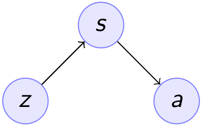
Time of arrival at work graphical model where T is a traffic jam and x_1 is the time John arrives at the office and x_2 is the time Jane arrives at the office.
*Conditional independence:

- Even though x_1, x_2 are not independent, they become independent once you know T.

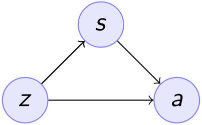
School admission

School	Male	Female
A	62	82
B	63	68
C	37	34
D	33	35
E	28	24
F	6	7

- z: gender
- s: school applied to
- a: admission

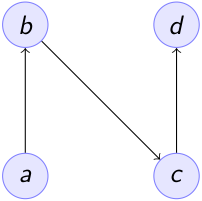


Is admission independent of gender?



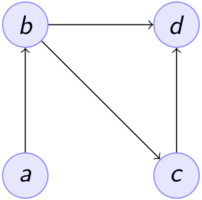
How about here?

What is the model for this graph?



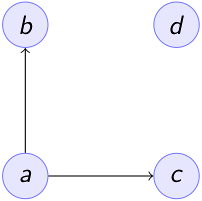
$$P(a, b, c, d) = \cdots$$

What is the model for this graph?



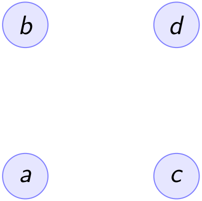
$$P(a, b, c, d) =$$

What is the model for this graph?



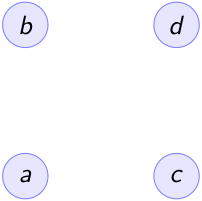
$$P(a, b, c, d) =$$

Draw the graph for this model



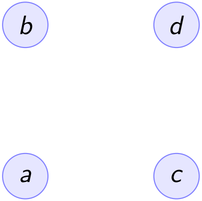
$$P(a, b, c, d) = P(a)P(b|a)P(c|b)P(d|b)$$

Draw the graph for this model



$$P(a, b, c, d) = P(a)P(b|a)P(d|c)P(c)$$

Draw the graph for this model



$$P(a, b, c, d) = P(a)P(b|a)P(c|a)P(d|b, c)$$

Generative modelling

General idea

- Data (x_t, y_t).
- Need to model $P(y|x)$.
- Model the complete data distribution: $P(x|y)$, $P(x)$, $P(y)$.
- Calculate $P(y|x) = \frac{P(x|y)P(x)}{P(y)}$.

Examples

- Naive Bayes classifier.
- Gaussian mixture model.
- Large language models.

Modelling the data distribution in classification

- Need to estimate the density $P(x|y)$ for each class y.
- Need to estimate $P(y)$.

The basic graphical model

A discriminative classification model

Here $P(y|x)$ is given directly. $y \longleftarrow x$

A generative classification model

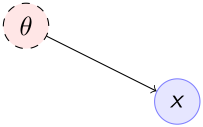
Here $P(y|x) = P(x|y)P(y)/P(x)$. $y \longrightarrow x$

An unsupervised generative model

Here we just have $P(x)$. x

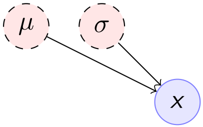
Adding parameters to the graphical model

A Bernoulli RV



Here, $x|\theta \sim \mathrm{Bernoulli}(\theta)$

A normally distributed variable



Here $x|\mu, \sigma \sim \mathrm{Normal}(\mu, \sigma^2)$

Classification: Naive Bayes Classifier

- Data (x, y)
- $x \in X$
- $y \in Y \subset \mathbb{N}$, N_i: amount of data from class i.

Separately model each class

- Assume each class data comes from a different normal distribution
- $x|y = i \sim \mathrm{Normal}(\mu_i, \sigma_i I)$
- For each class, calculate
  - Empirical mean $\hat{\mu}_i = \sum_{t:y_t=i} x_t / N_i$
  - Empirical variance $\hat{\sigma}_i$.

Decision rule

Use Bayes's theorem:

$$P(y|x) = P(x|y)P(y)/P(x),$$

choosing the y with largest posterior $P(y|x)$.

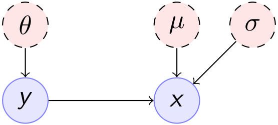
- $P(x|y = i) \propto \exp(-\|\hat{\mu}_i - x\|^2/\hat{\sigma}_i^2)$

Graphical model for the Naive Bayes Classifier

When $x \in \mathbb{R}$

Assume k classes, then

- $\mu = (\mu_1, \ldots, \mu_k)$
- $\sigma = (\sigma_1, \ldots, \sigma_k)$
- $\theta = (\theta_1, \ldots, \theta_k)$



- $y \mid \theta \sim \mathrm{Mult}(\theta)$
- $x \mid y, \mu, \sigma \sim \mathrm{Normal}(\mu_y, \sigma_y^2)$

General idea

Parametric models

- Fixed histograms
- Gaussian Mixtures

Non-parametric models

- Variable-bin histograms
- Infinite Gaussian Mixture Model
- Kernel methods

Histograms

Fixed histogram

- Hyper-Parameters: number of bins
- Parameters: Number of points in each bin.

Variable histogram

- Hyper-parameters: Rule for constructing bins
- Generally $\sqrt{n}$ points in each bin.

Gaussian Mixture Model

Hyperparameters:

- Number of Gaussian k.

Parameters:

- Multinomial distribution $\boldsymbol{\theta}$ over Gaussians
- For each Gaussian i, center μ_i, covariance matrix Σ_i.

Algorithms:

- Expectation Maximisation
- Gradient Ascent
- Variational Bayesian Inference (with appropriate prior)

Details of Gaussian mixture models

Model. For each point x_t:

- $z_t \mid \theta \sim \mathrm{Mult}(\theta_i)$, $\theta \in \mathbb{\Delta}^k$
- $x_t | z_t = i \sim \mathrm{Normal}(\mu_i, \Sigma_i)$.
- $\mathrm{Mult}(\theta)$ is multinomial

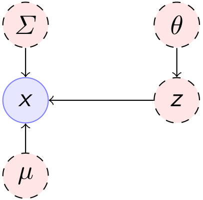
$$\mathbb{P}(z_t = i \mid \theta) = \theta_i$$

- $\mathrm{Normal}(\mu, \Sigma)$ is multivariate Gaussian

$$p(x \mid \mu, \Sigma) \propto \exp(-\frac{1}{2}(x - \mu)^\top \Sigma^{-1}(x - \mu))$$

- The generating distribution is

$$p(x|\theta, \mu, \Sigma) = \sum_{z \in [k]} p(x \mid \mu_z, \Sigma_z) P(z \mid \theta).$$



Gradient ascent

In the following we use θ for all the parameters of the Gaussian mixture model, with $x = (x_1, \ldots, x_T)$ and $z = (z_1, \ldots, z_T)$

Objective function

One way to estimate θ is through maximising the likelihood $L(\theta) = P(x|\theta)$

Marginalisation over latent variable

However, we need to marginalise over all values z

$$L(\theta) = \sum_z P(z, x|\theta)$$

For T data points and k different values of z_t, there are k^T vectors z to sum over.

Gradient ascent

If we can calculate the gradient of L, we can use gradient ascent to update our parameters:

$$\theta^{(n+1)} = \theta^{(n)} + \alpha \nabla_\theta L(\theta).$$

Gradient calculation

Here we use the log trick: $\nabla \ln f(\theta) = \nabla f(\theta)/f(\theta)$.

$$
\begin{aligned}
\nabla_\theta L(\theta) &= \sum_z \nabla_\theta P(z, x \mid \theta) && (6)\\
&= \sum_z P(z, x \mid \theta) \nabla_\theta \ln P(z, x \mid \theta) && (7)\\
&= \sum_z P(x \mid z, \theta) P(z \mid \theta) \nabla_\theta \ln P(z, x \mid \theta) && (8)\\
&\approx \frac{1}{m} \sum_{i=1}^m P(x \mid z^{(i)}, \theta) \nabla_\theta \ln P(z^{(i)}, x \mid \theta) \qquad z^{(i)} \sim P(z \mid \theta) && (9)
\end{aligned}
$$

The final approximates the sum with the sample mean, sampling $z^{(i)}$ from the distribution. Hence, we can implement the following algorithm

- For $i = 1, \ldots, m$: $z^{(i)} \sim P(z \mid \theta^{(n)})$
- $d^{(n)} = \frac{1}{m} \sum_{i=1}^m P(x \mid z^{(i)}, \theta) \nabla_\theta \ln P(z^{(i)}, x \mid \theta^{(n)})$
- $\theta^{(n+1)} = \theta^n + \alpha d^{(n)}$.

A lower bound on the likelihood

For any distribution $G(z)$, and specifically for $G(z) = P(z|x, \theta^{(k)})$:

$$\begin{aligned}
\ln P(x|\theta) &= \sum_z G(z) \ln P(x|\theta) = \sum_z G(z) \ln[P(x, z|\theta)/P(z|x, \theta)] \\
&= \sum_z G(z)[\ln P(x, z|\theta) - \ln P(z|x, \theta)] \\
&= \sum_z G(z) \ln P(x, z|\theta) - \sum_z G(z) \ln P(z|x, \theta) \\
&= \sum_z P(z|x, \theta^{(k)}) \ln P(x, z|\theta) - \sum_z P(z|x, \theta^{(k)}) \ln P(z|x, \theta) \\
&\geq \sum_z P(z|x, \theta^{(k)}) \ln P(x, z|\theta) - \sum_z P(z|x, \theta^{(k)}) \ln P(z|x, \theta^{(k)}) \\
&= Q(\theta \mid \theta^{(k)}) + \mathbb{H}(z \mid x, \theta^{(k)}),
\end{aligned}$$

where

$$\mathbb{H}(z \mid x, \theta^{(k)}) = \sum_z P(z \mid x, \theta^{(k)}) \ln P(z \mid x, \theta^{(k)})$$

is the entropy of z for a fixed $x, \theta^{(k)}$. As this is not negative, $\ln P(x|\theta) \geq Q(\theta \mid \theta^{(k)})$.

Some information theory

Information theory notation can be a bit confusing. Sometimes we talk about random variables ω, and sometimes about probability measures P. This is context-dependent.

Entropy

For a random variable ω under distribution P, we denote the entropy as

$$\mathbb{H}_P(\omega) \equiv \mathbb{H}(P) \equiv \mathbb{H}(\omega) = \sum_{\omega \in \Omega} P(\omega) \ln P(\omega).$$

KL Divergence

For two probabilities P, Q over random outcomes in the same space Ω, we define

$$D_{KL}(P \| Q) = \sum_{\omega \in \Omega} P(\omega) \ln \frac{P(\omega)}{Q(\omega)}$$

The Gibbs Inequality

$D_{KL}(P \| Q) \geq 0$, or $\sum_x \ln P(x) P(x) \geq \sum_x \ln Q(x) P(x)$.

EM Algorithm (Dempster et al, 1977)

- Initial parameter $\boldsymbol{\theta}^{(0)}$, observed data x
- For $k = 0, 1, \ldots$

– Expectation step:

$$Q(\boldsymbol{\theta} \mid \boldsymbol{\theta}^{(k)}) \triangleq \mathbb{E}_{z \sim P(z|x,\boldsymbol{\theta}^{(k)})}[\ln P(x, z|\boldsymbol{\theta})] = \sum_{z} [\ln P(x, z|\boldsymbol{\theta})] P(z \mid x, \boldsymbol{\theta}^{(k)})$$

– Maximisation step:

$$\boldsymbol{\theta}^{(k+1)} = \arg\max_{\boldsymbol{\theta}} Q(\boldsymbol{\theta}, \boldsymbol{\theta}^{(k)}).$$

See *Expectation-Maximization as lower bound maximization, Minka, 1998*

Minorise-Maximise

EM can be seen as a version of the minorise-maximise algorithm

- $f(\boldsymbol{\theta})$: Target function to maximise
- $Q(\boldsymbol{\theta}|\boldsymbol{\theta}^{(k)})$: surrogate function

Q Minorizes f

This means surrogate is always a lower bound so that

$$f(\boldsymbol{\theta}) \geq Q(\boldsymbol{\theta}|\boldsymbol{\theta}^{(k)}), \qquad f(\boldsymbol{\theta}^{(k)}) \geq Q(\boldsymbol{\theta}^{(k)}|\boldsymbol{\theta}^{(k)}),$$

Algorithm

- Calculate: $Q(\boldsymbol{\theta}|\boldsymbol{\theta}^{(k)})$
- Optimise: $\boldsymbol{\theta}^{(k+1)} = \arg\max_{\boldsymbol{\theta}} Q(\boldsymbol{\theta}|\boldsymbol{\theta}^{(k)})$.

GMM versus histogram

- Generate some data x from an arbitrary distribution in $\mathbb{R}$.
- Fit the data with a histogram for varying numbers of bins
- Fit a GMM with varying numbers of Gaussians
- What is the best fit? How can you measure it?

GMM Classifier

Base class: sklearn GaussianMixtureModel

- *fit()* only works for Density Estimaiton
- *predict()* only predicts cluster labels

Problem

- Create a GMMClassifier class
- *fit()* should take X, y, arguments
- *predict()* should predict class labels
- Hint: Use $predict_{proba}()$ and multiple GMM models