Learning with fully observed variables

Generative and Deep Learning (GDL)

Davide Bacciu (davide.bacciu@unipi.it)

UNIVERSITÀ DI PISA

Lecture Outline

- Introduction to the module
- Learning as an inference problem
- Flavors of probabilistic learning
- (Maximum Likelihood) Learning with fully observed variables
- Learning the Naïve Bayes classifier

Module II – Learning in probabilistic models

Lesson 1 Learning with fully observed variables

Lesson 2 Learning with hidden variables

Lesson 3 HMM I – Model and forward-backward

Lesson 4 HMM II – Learning and inference with hidden variables

Lesson 5 Variational approximation and Generalized Expectation maximization

Lesson 6 Latent Dirichlet Allocation

Lesson 7 Sampling methods

Lesson 8 Undirected models: MRF and Boltzmann machines

Module content is almost entirely covered by David Barber's book

Learning in Probabilistic Models

Previously: Probabilistic Inference

Given a set of available hypotheses $h \in H$ (possibly infinite) and some evidence $\boldsymbol{d}$, we can answer a probabilistic query on X (inference) in three possible ways

- Bayesian - $P(X|\boldsymbol{d}) = \sum_i P(X|h_i)P(h_i|\boldsymbol{d})$
- Maximum a-Posteriori (MAP) - $h_{MAP} = \arg\max_{h \in H} P(h|\boldsymbol{d}) = \arg\max_{h \in H} P(\boldsymbol{d}|h)P(h)$
- Maximum Likelihood (ML) - $h_{ML} = \arg\max_{h \in H} P(\boldsymbol{d}|h)$

- Probabilistic learning is about finding the parameters θ of a distribution $\mathrm{P}_\theta(\cdot)$ (or equivalently $P(\cdot|\theta)$)
- Can be casted as the inference problem of identifying which parameterized hypothesis h_θ to use
  - h_θ is our learning model, whose properties depend on the nature of the family of distributions $\mathrm{P}_\theta(\cdot)$
  - family of distributions: Bernoulli, Gaussian, Gamma, Dirichlet, …
- So: learning can be pursued by Bayesian, MAP or ML inference

Considerations Bayesian Learning

Bayesian inference has clear tractability iussues

$$P(X|\boldsymbol{d}) = \int_{\theta} P(X|h_\theta)P(h_\theta|\boldsymbol{d})d\theta$$

- ML and MAP are point estimates of the Bayesian since they infer based only on one most likely hypothesis $h_{\theta\,MAP}$ or $\mathrm{h}_{\theta_{ML}}$
- MAP is a regularization of the ML estimation
  - Hypothesis prior $P(h_\theta)$ embodies trade-off between complexity and degree of fit
  - Well-suited to working with small datasets and/or large parameter spaces

Prior Regularization

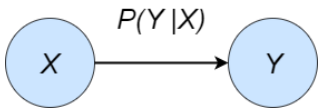
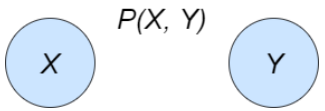
- $P(h_\theta)$ introduces preference across hypotheses
- Penalize complexity
  - Complex hypotheses have a lower prior probability
  - Hypothesis prior embodies trade-off between complexity and degree of fit
- MAP hypothesis h_{MAP}

$$\max_{h_\theta} P(\boldsymbol{d}|h_\theta)P(h_\theta) \equiv \min_{h} -\log_2(P(\boldsymbol{d}|h_\theta)) - \log_2 P(h_\theta)$$

Number of bits required to specify h_θ

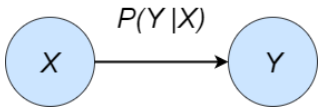
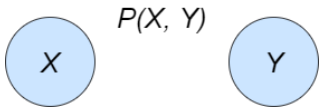
- MAP $\Longrightarrow$ choosing the hypothesis that provides maximum compression
- MAP is a regularization of the ML estimation

Learning with graphical models

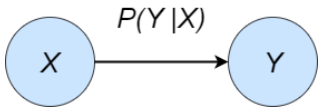
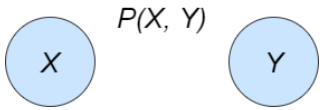
		Structure	
		Fixed Structure 	Fixed Variables 
Data	Complete	Naive Bayes Calculate Frequencies (ML)	Discover dependencies from the data Structure Search Independence tests
	Incomplete	Latent variables EM Algorithm (ML) MCMC, VBEM (Bayesian)	Difficult Problem Structural EM

Learning with graphical models

Today!

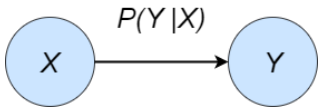
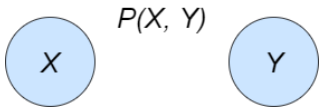
		Structure	
		Fixed Structure 	Fixed Variables 
Data	Complete	**Naive Bayes** **Calculate Frequencies (ML)**	Discover dependencies from the data Structure Search Independence tests
	Incomplete	Latent variables EM Algorithm (ML) MCMC, VBEM (Bayesian)	Difficult Problem Structural EM

Learning with graphical models

		Structure	
		Fixed Structure 	Fixed Variables 
Data	Complete	Naive Bayes Calculate Frequencies (ML)	Discover dependencies from the data Structure Search Independence tests
	Incomplete	**Latent variables** **EM Algorithm (ML)** **MCMC, VBEM (Bayesian)**	Difficult Problem Structural EM

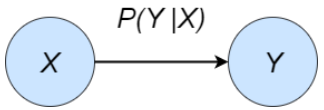
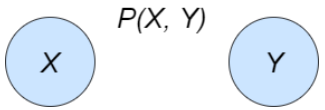
Next lecture(s)!

Learning with graphical models

		Structure	
		Fixed Structure 	Fixed Variables 
Data	Complete	Naive Bayes Calculate Frequencies (ML)	**Discover dependencies from the data Structure Search Independence tests**
	Incomplete	Latent variables EM Algorithm (ML) MCMC, VBEM (Bayesian)	Difficult Problem Structural EM

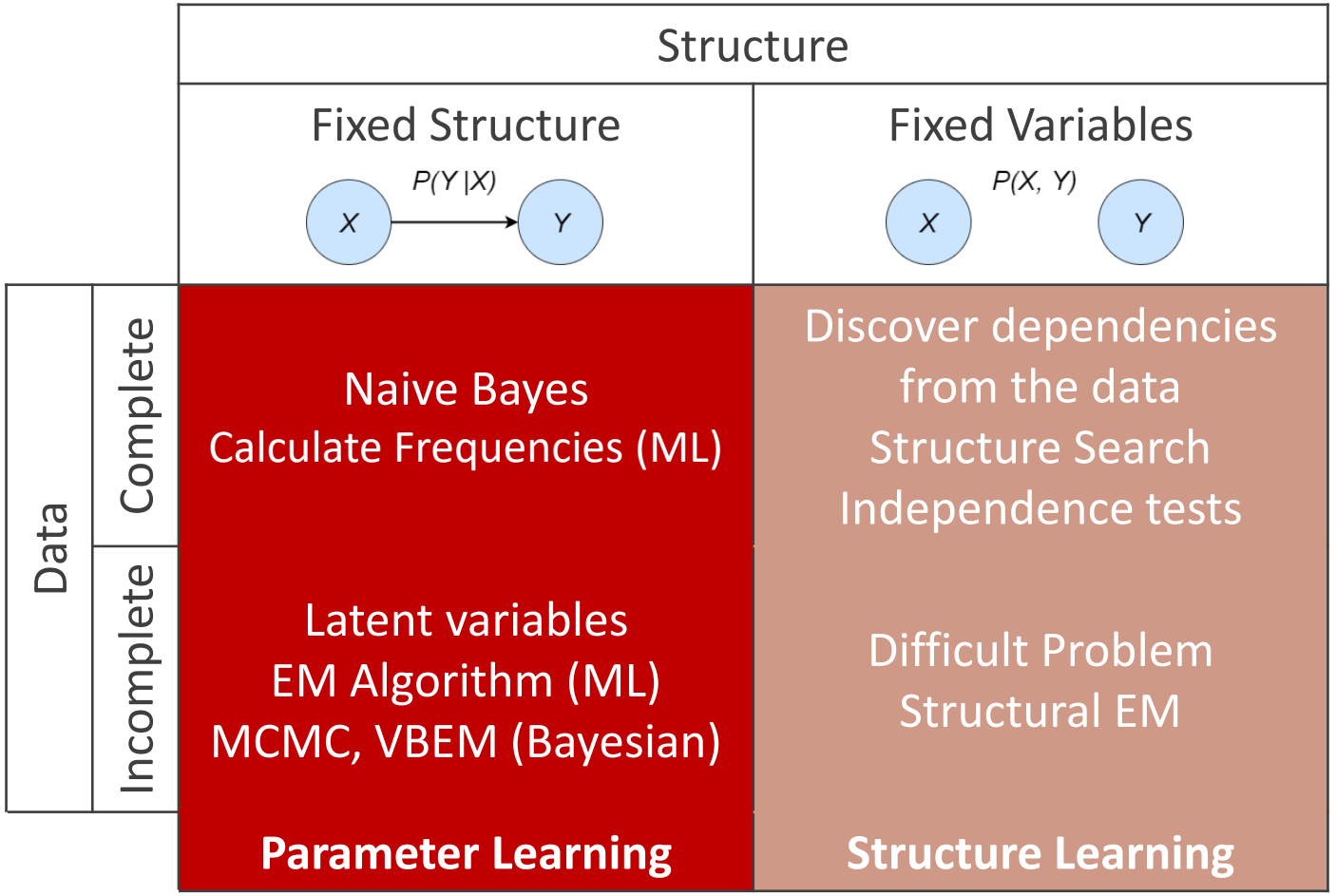
Last lecture!

Learning with graphical models

		Structure	
		Fixed Structure 	Fixed Variables 
Data	Complete	Naive Bayes Calculate Frequencies (ML)	Discover dependencies from the data Structure Search Independence tests
	Incomplete	Latent variables EM Algorithm (ML) MCMC, VBEM (Bayesian)	**Difficult Problem** **Structural EM**

Not in this course!
(luckily for you?)

Learning with graphical models



Parameter learning with fully observable models

Parameter Learning with Complete Data

- Find numerical parameters for a graphical model with known structure and observables for all random variables involved
- Determine the best hypothesis h_θ regulated by a (set of) parameter θ

  h_θ : the expected proportion of coin tosses returning heads is θ
- With the usual Bayesian probabilities

$$\text{Prior} \quad P(h_\theta) = P(\theta)$$
$$\text{Likelihood} \quad P(\boldsymbol{d}|h_\theta) = P(\boldsymbol{d}|\theta)$$
$$\text{Posterior} \quad P(h_\theta|\boldsymbol{d}) = P(\theta|\boldsymbol{d})$$

- If hypotheses are equiprobable, it is reasonable to try Maximum Likelihood Estimation (MLE)

Maximum-Likelihood (ML) Learning

Find the model θ that is most likely to have generated the data $\boldsymbol{d} = \{\boldsymbol{x_1}, \dots, \boldsymbol{x_N}\}$, assuming data is independently and identically distributed (i.i.d)

$$\theta_{ML} = \arg\max_{\theta \in \Theta} P(\boldsymbol{d}|\theta)$$

from a family of parameterized distributions $P(x|\theta)$.

Maximum-Likelihood (ML) Learning

Find the model θ that is most likely to have generated the data $\boldsymbol{d} = \{\boldsymbol{x_1}, \dots, \boldsymbol{x_N}\}$, assuming data is independently and identically distributed (i.i.d)

$$\theta_{ML} = \arg\max_{\theta \in \Theta} P(\boldsymbol{d}|\theta) = \arg\max_{\theta \in \Theta} P(\boldsymbol{x_1}, \dots, \boldsymbol{x_N}|\theta)$$

from a family of parameterized distributions $P(x|\theta)$.

Maximum-Likelihood (ML) Learning

Find the model θ that is most likely to have generated the data $\boldsymbol{d} = \{\boldsymbol{x_1}, \dots, \boldsymbol{x_N}\}$, assuming data is independently and identically distributed (i.i.d)

$$\theta_{ML} = \arg\max_{\theta \in \Theta} P(\boldsymbol{d}|\theta) = \arg\max_{\theta \in \Theta} P(\boldsymbol{x_1}, \dots, \boldsymbol{x_N}|\theta) = \arg\max_{\theta \in \Theta} \prod_{i=1}^{N} P(\boldsymbol{x_i}|\theta)$$

from a family of parameterized distributions $P(x|\theta)$.

Optimization problem that considers the Likelihood function

$$\mathcal{L}(\theta|x) = P(x|\theta)$$

to be a function of θ.

Can be addressed by solving $\frac{\partial \mathcal{L}(\theta|x)}{\partial \theta} = 0$

Example – The Infamous Biased Coin (MLE)

- The outcome of a coin toss can be modelled as random discrete variable C

  $P(C = head) = \theta \qquad P(C = tail) = 1 - \theta$ (Bernoulli Distribution)

- By varying the value of $\theta \in [0,1]$, we obtain different models (parametric hypothesis space)
- We can measure the "goodness" of a model h_θ by asking ourselves: which is the probability of observing what we have seen if we use h_θ to describe the world?
- This is the likelihood: the probability that an event occurs

Example – The Infamous Biased Coin (MLE)

- Assume observed data $\boldsymbol{d}$ are independent and identically distributed (i.i.d), how likely is to observe n_H H and n_T T over n tosses if the $\mathrm{P}(head) = \theta$?

$$P(\boldsymbol{d} \mid \theta) = \theta^{n_H}(1-\theta)^{n_T}$$

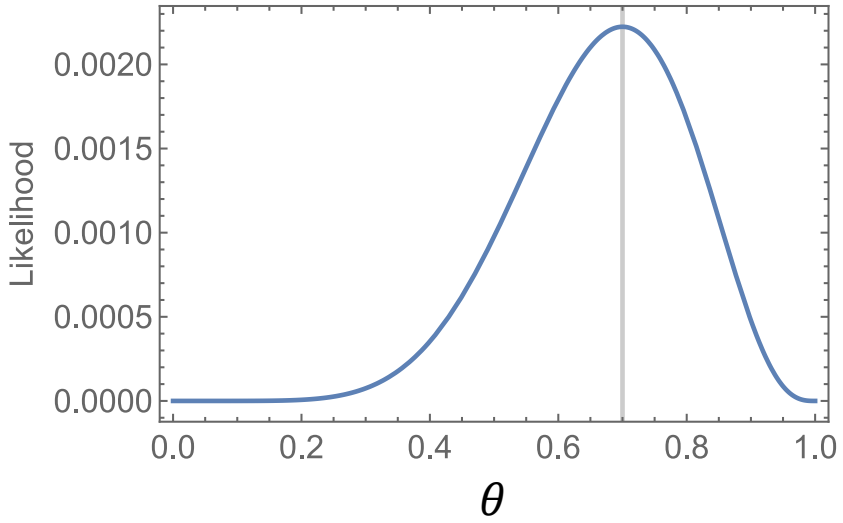
- ML learning means finding the parameter θ_{ML} s.t.

$$\theta_{ML} = \underset{\theta \in [0,1]}{\operatorname{argmax}} P(\boldsymbol{d} \mid \theta) \text{ or better } \theta_{ML} = \underset{\theta \in [0,1]}{\operatorname{argmax}} \log P(\boldsymbol{d} \mid \theta)$$

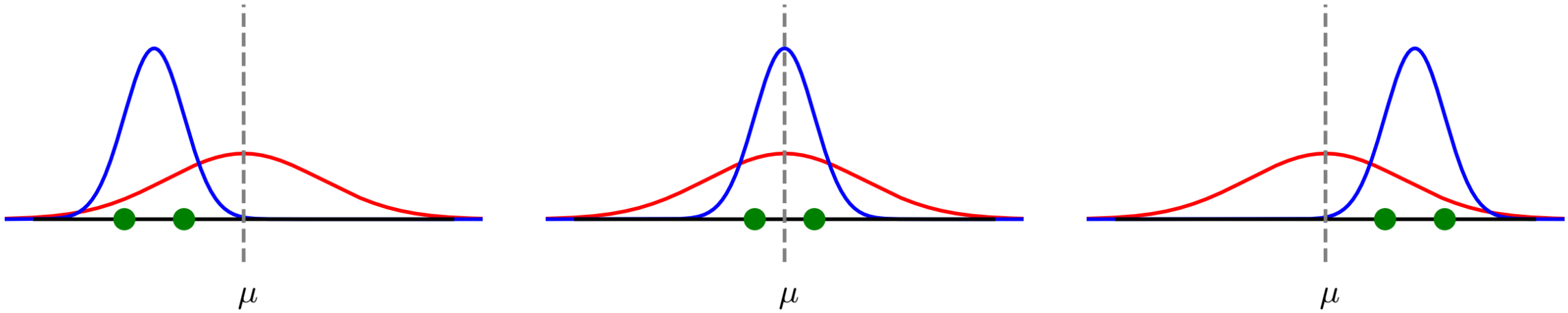
- And the winner ML parameter θ_{ML} is…

$$\theta_{ML} = \frac{n_H}{n_H + n_T} = \frac{n_H}{n}$$

MLE is entirely data-driven: we cannot encode our prior belief (e.g. about bias!)



ML vs MAP



The Coin - MAP Edition

Treat the **parameters $\boldsymbol{\theta}$ as a random variable** and define a prior distribution $P(\theta)$ encoding our beliefs (e.g. of the coin being biased towards heads)

Estimate our model using the **MAP hypothesis**

$$\theta_{MAP} = \underset{\theta \in [0,1]}{\operatorname{argmax}} \log P(\boldsymbol{d} \mid \theta) P(\theta)$$

where the posterior is the **distribution for the expected proportion of heads $\boldsymbol{\theta}$** given the data

We need a distribution for the prior (and the posterior)!

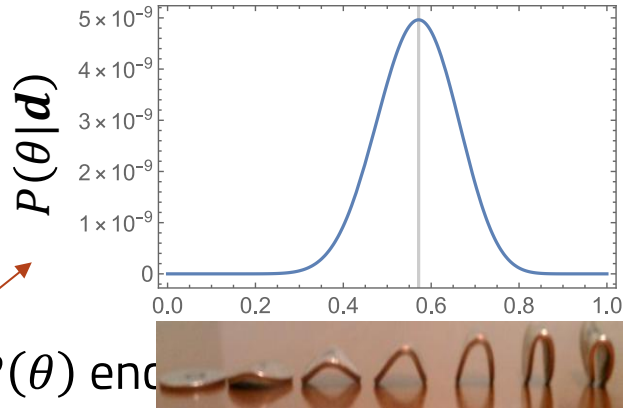
- Luckily, we learned about conjugate distributions so we know that for a Bernoulli likelihood we can use a Beta prior

$$P(\theta|\boldsymbol{d}) = Beta(\alpha + n_H, \beta + n_T) = \frac{1}{B(\alpha + n_H, \beta + n_T)} \theta^{\alpha-1}(1-\theta)^{\beta-1} \theta^{n_H}(1-\theta)^{n_T}$$

$$P(\theta|\boldsymbol{d}) \propto \theta^{\alpha-1+n_H}(1-\theta)^{\beta-1+n_T}$$

- α and β can be thought of as imaginary counts of our prior experience
- We can discard the normalization constant $B(\cdot,\cdot)$ as it will not impact MAP optimization

The Coin - MAP Edition



Treat the **parameters θ as a random variable** and define a prior distribution $P(\theta)$ enc beliefs (e.g. of the coin being biased towards heads)

Estimate our model using the **MAP hypothesis**

$$\theta_{MAP} = \underset{\theta\in[0,1]}{\text{argmax}} \log P(\boldsymbol{d} \mid \theta)P(\theta)$$

where the posterior is the **distribution for the expected proportion of heads θ** given the data

We need a distribution for the prior (and the posterior)!

- Luckily, we learned about conjugate distributions so we know that for a Bernoulli likelihood we can use a Beta prior

$$P(\theta|\boldsymbol{d}) = Beta(\alpha + n_H, \beta + n_T) = \frac{1}{B(\alpha + n_H, \beta + n_T)}\theta^{\alpha-1}(1-\theta)^{\beta-1}\theta^{n_H}(1-\theta)^{n_T}$$

$$P(\theta|\boldsymbol{d}) \propto \theta^{\alpha-1+n_H}(1-\theta)^{\beta-1+n_T}$$

- α and β can be thought of as imaginary counts of our prior experience
- We can discard the normalization constant $B(\cdot,\cdot)$ as it will not impact MAP optimization

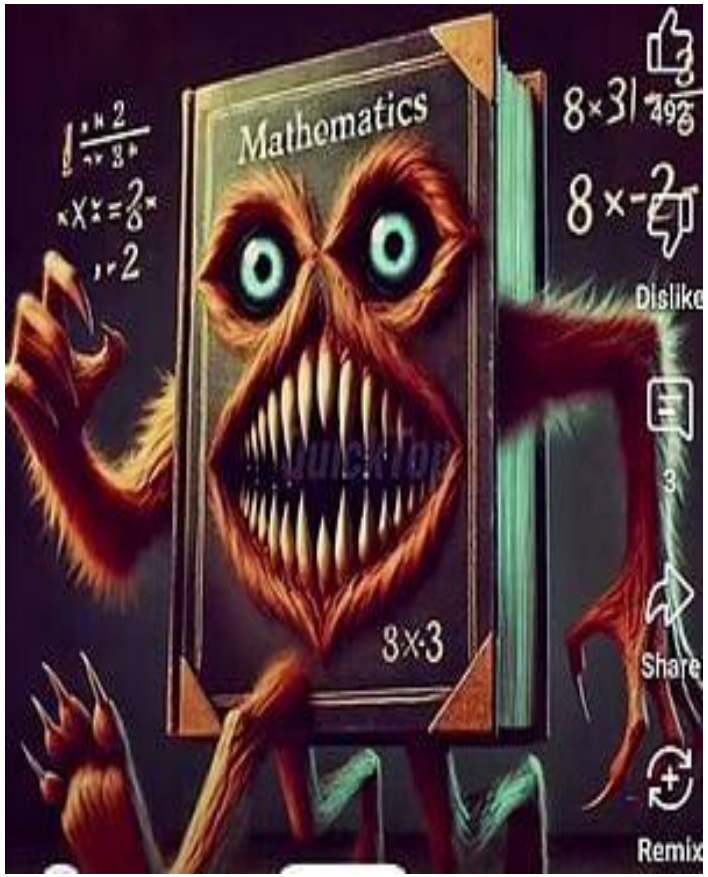
The Bayesian Coin

Bayesian learning allows to use the whole posterior (and not a point estimate) to perform predictions

$$P(x'|\boldsymbol{d}) = \int_{\theta} P(x'|\theta)P(\theta|\boldsymbol{d})d\theta$$

- We do not learn a single model, but a distribution over all the possible models

The Bayesian Coin



Bayesian learning allows to use the whole posterior (and not a point estimate) to perform predictions

$$P(x'|\boldsymbol{d}) = \int_{\theta} P(x'|\theta)P(\theta|\boldsymbol{d})d\theta$$

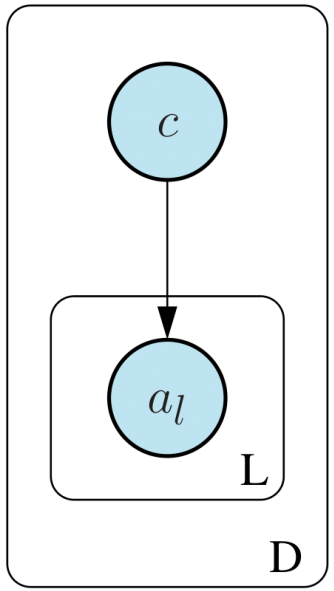
- We do not learn a single model, but a distribution over all the possible models
- …and this is where your nightmares begin

We cannot disregard the normalizing constant in the Beta anymore

$$Beta(\alpha + n_H, \beta + n_T) = \frac{\theta^{\alpha-1+n_H}(1-\theta)^{\beta-1+n_T}}{B(\alpha + n_H, \beta + n_T)}$$

Things ($P(x'|\boldsymbol{d})$) become analytically intractable

Naive Bayes Classifier



One of the simplest probabilistic classifier based on a strong independence assumption

Consider the setting

- Target classification function $f: X \longrightarrow C$
- Each instance $x \in X$ is described by a set of attributes (discrete-valued to ease presentation)

$$x = < a_1, \ldots, a_l, \ldots, a_L >$$

- Seek the MAP classification

$$c_{NB} = \arg\max_{c_j \in C} P(c_j | a_1, \ldots, a_L)$$

Naive Bayes Assumption

Rewrite posterior using the likelihood and apply the Naïve assumption

$$c_{NB} = \arg\max_{c_j \in C} P(c_j | a_1, \ldots, a_L)$$

$$= \arg\max_{c_j \in C} P(a_1, \ldots, a_L | c_j) P(c_j) = \arg\max_{c_j \in C} \prod_{l=1}^{L} P(a_l | c_j)\, P(c_j)$$

Conditional independence between the attributes al given classification c_j

Now we can run Maximum Likelihood estimation of the NB parameters

- $P(a_l = s | c = k)$ s.t. $1 \leq s \leq S$ and $1 \leq k \leq K$
- $P(c = k)$ s.t. $1 \leq k \leq K$

These are multinomial probability tables and ML learning fills their values

Naive Bayes Likelihood

Given N observed training pairs (x_j, c_j) s.t. $x_j =< a_1, \ldots, a_L >$

$$\mathcal{L}(\theta|\boldsymbol{d}) = P(\boldsymbol{d}|\theta) = \prod_{j=1}^{D} P(c_j) \prod_{l=1}^{L} P(a_{lj}|c_j)$$

$$= \prod_{j=1}^{D} \prod_{k=1}^{K} P(c = k)^{z_{jk}} \left(\prod_{l=1}^{L} \prod_{s=1}^{S} \left(P(a_l = s|c = k) \right)^{t_j^{ls}} \right)^{z_{jk}}$$

Indicator variables

$$\begin{cases} z_{jk} = \begin{cases} 1 \text{ if } x_j \text{ is classified } k \\ 0 \text{ otherwise} \end{cases} \\ t_j^{ls} = \begin{cases} 1 \text{ if } a_{jl} \text{ has value } s \\ 0 \text{ otherwise} \end{cases} \end{cases}$$

Maximization of Naive Bayes Log-Likelihood

Maximize w.r.t. parameters $\theta =< P(c = k), P(a_l = s|c = k) >$
subject to $\sum_k P(c = k) = 1$ and $\sum_s P(a_l = s|c) = 1$

$$\log(\mathcal{L}(\theta|\boldsymbol{d})) = \sum_{j=1}^{D}\sum_{k=1}^{K} z_{jk} \log P(c = k) + \sum_{j=1}^{D}\sum_{k=1}^{K} z_{jk} \left(\sum_{l=1}^{L}\sum_{s=1}^{S} t_j^{ls} \log P(a_l = s|c = k) \right)$$

yields

$$P(c = k) = \frac{\sum_{j=1}^{D} z_{jk}}{D} = \frac{N(k)}{D}$$

$$P(a_l = s|c = k) = \frac{\sum_{j=1}^{D} z_{jk}\, t_j^{ls}}{\sum_{j=1}^{D}\sum_{s=1}^{S^l} z_{jk}\, t_j^{ls}} = \frac{N_{ls}(k)}{\sum_{s=1}^{S_l} N_{ls}(k)}$$

Naive Bayes Classification

Need to be careful when applying ML to **scarce data**

- Learning essentially amounts to counting frequencies
- What happens if a class c_k has no occurrences of an attribute $a_l = s$?

$$P(a_l = s|c = k) = \frac{N_{ls}(k)}{\sum_{s=1}^{S_l} N_{ls}(k)} = 0 \Longrightarrow P(c = k|x) = 0 \;\; \forall x$$

- Add a constant term α in both the numerator and the denominator to smooth the estimation (Laplacian smoothing)

$$P(a_l = s|c = k) = \frac{N_{ls}(k) + \alpha}{\sum_{s=1}^{S_l}(N_{ls}(k) + \alpha)}$$

- α is a priori estimate of the attribute-class probability
  - Consider $\beta = P(a_l = s|c = k)$ as a random variable
  - Prior distribution $P(\beta|\alpha)$ with hyperparameter α

Take Home Messages

- Maximum Likelihood can be used for parameter learning (density estimation) when all random variables are observable
- For discrete distributions Maximum Likelihood reduces to counting
  - Frequentist approach
  - Estimates can degenerate if there is insufficient data support
- Smoothing helps regularizing estimates (MAP approach)
  - Amounts to defining suitable priors
  - Virtual counts
- Bayesian learning is powerful, but doesn't come for free

Next Lecture

Discovering what happens when not all random variables are observable

- Latent/hidden variable models
- Maximum likelihood learning with non-observables
- (Exact) Expectation-Maximization
- Gaussian mixture model