Ekaterina Lobacheva

Introduction to Bayesian methods

2021

Slides are partially based on lectures of Dmitry Vetrov, Dmitry Kropotov and Kirill Struminsky, deepbayes.ru/2018

Section 3: Bayesian Deep Learning

Part 1. Intro to Bayesian Methods

Part 2. Bayesian Neural Networks

Part 3. Variational Autoencoders

Section 3: Bayesian Deep Learning

Part 1. Intro to Bayesian Methods

Part 2. Bayesian Neural Networks

Part 3. Variational Autoencoders

In this video: Bayesian Framework and Bayesian ML Models

How to work with distributions?

$$\text{Conditional} = \frac{\text{Joint}}{\text{Marginal}}, \quad p(x|y) = \frac{p(x,y)}{p(y)}$$

Product rule

any joint distribution can be expressed as a product of one-dimensional conditional distributions

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

Sum rule

any marginal distribution can be obtained from the joint distribution by integrating out unnecessary variables

$$p(y) = \int p(x, y)dx$$

Example

- We have a joint distribution over three groups of variables $p(x, y, z)$
- We observe x and are interested in predicting y
- Values of z are unknown and irrelevant to us
- How to estimate $p(y|x)$ from $p(x, y, z)$?

Example

- We have a joint distribution over three groups of variables $p(x, y, z)$
- We observe x and are interested in predicting y
- Values of z are unknown and irrelevant to us
- How to estimate $p(y|x)$ from $p(x, y, z)$?

$$p(y|x) = \frac{p(x, y)}{p(x)} = \frac{\int p(x, y, z)dz}{\int p(x, y, z)dzdy}$$

Sum rule and product rule allow to obtain arbitrary conditional distributions from the joint one

Bayes theorem

Bayes theorem (follows from product and sum rules):

$$p(y|x) = \frac{p(x,y)}{p(x)} = \frac{p(x|y)p(y)}{p(x)} = \frac{p(x|y)p(y)}{\int p(x|y)p(y)dy}$$

Bayes theorem defines the rule for uncertainty conversion when new information arrives:

$$\text{Posterior} \; = \frac{\text{Likelihood} \; \times \; \text{Prior}}{\text{Evidence}}$$

Statistical inference

Problem: given i.i.d. data $X = (x_1, ..., x_n)$ from distribution $p(x|\theta)$ one needs to estimate θ

Frequentist framework: use maximum likelihood estimation (MLE)

$$\theta_{ML} = \arg\max p(X|\theta) = \arg\max \prod_{i=1}^{n} p\left(x_i|\theta\right) = \arg\max \sum_{i=1}^{n} \log p\left(x_i|\theta\right)$$

Bayesian framework: encode uncertainty about θ in a prior $p(\theta)$ and apply Bayesian inference

$$p(\theta|X) = \frac{\prod_{i=1}^{n} p\left(x_i|\theta\right) p(\theta)}{\int \prod_{i=1}^{n} p\left(x_i|\theta\right) p(\theta) d\theta}$$

Example: coin tossing

- We have a coin which may be fair or not
- The task is to estimate a probability θ of landing heads up
- Data: 2 tosses with a result (H,H)

Head (H)

Tail (T)

Example: coin tossing

- We have a coin which may be fair or not
- The task is to estimate a probability θ of landing heads up
- Data: 2 tosses with a result (H,H)

Head (H) Tail (T)

Frequentist framework:

In all experiments the coin landed heads up
$\theta_{ML} = 1$

→

The coin is not fair and always lands heads up

Example: coin tossing

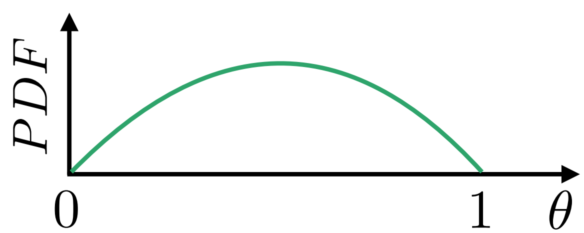
- We have a coin which may be fair or not
- The task is to estimate a probability θ of landing heads up
- Data: 2 tosses with a result (H,H)

Head (H)

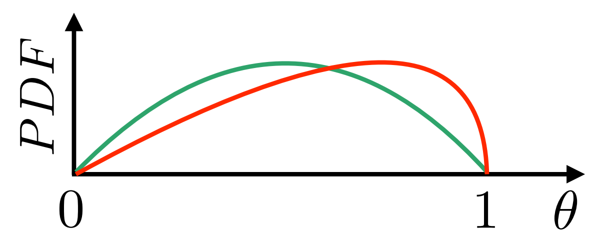
Tail (T)

Bayesian framework:

Prior



Posterior



Example: coin tossing

- We have a coin which may be fair or not
- The task is to estimate a probability θ of landing heads up
- Data: 1000 tosses with a result (H,H,T,…) — 489 tails and 511 heads

Head (H)

Tail (T)

Example: coin tossing

- We have a coin which may be fair or not
- The task is to estimate a probability θ of landing heads up
- Data: 1000 tosses with a result (H,H,T,…) — 489 tails and 511 heads

Head (H)

Tail (T)

Both frameworks:

Sufficient amount of data matches our expectations $\rightarrow$ The coin is fair

Frequentist vs. Bayesian frameworks

- Applicability of frequentist framework: $n >> d$
- Applicability of Bayesian framework: $\forall n$

- The number of tunable parameters in modern ML models is comparable with the sizes of training data

- Frequentist framework is a limit case of Bayesian one:

$$\lim_{n/d \to \infty} p\left(\theta | x_1, \ldots, x_n\right) = \delta\left(\theta - \theta_{ML}\right)$$

Bayesian framework just provides an alternative point of view, it DOES NOT contradict or deny frequentist framework

Advantages of Bayesian framework

- We can encode our prior knowledge or desired properties of the final solution into a prior distribution
- Prior is a form of regularization
- Additionally to the point estimate of θ posterior contains information about the uncertainty of the estimate

Probabilistic ML model

For each object in the data:

- x — set of observed variables (features)
- y — set of hidden / latent variables (class label / hidden representation, etc.)

Model:

- θ — model parameters (e.g. weights of the linear model)

Generative probabilistic ML model

Models joint distribution
$p(x, y, \theta) = p(x, y \mid \theta)p(\theta)$

$\Rightarrow$

Can generate new objects,
i.e. pairs (x, y)

May be quite difficult to train since the observed space is usually much more complicated than the hidden one

Examples:

- Generation of text, speech, images, etc.

Discriminative probabilistic ML model

Models $p(y, \theta \mid x)$ → Cannot generate new objects — needs x as an input

Usually assumes that prior over θ does not depend on x:

$$p(y, \theta \mid x) = p(y \mid x, \theta) p(\theta)$$

Examples:

- Classification or regression task (hidden space is much easier than the observed one)
- Machine translation (hidden and observed spaces have the same complexity)

Training Bayesian ML models

We are given training data (X_{tr}, Y_{tr}) and a discriminative model $p(y, \theta \mid x)$

Training stage — Bayesian inference over θ:

$$p\left(\theta \mid X_{tr}, Y_{tr}\right) = \frac{p\left(Y_{tr} \mid X_{tr}, \theta\right) p(\theta)}{\int p\left(Y_{tr} \mid X_{tr}, \theta\right) p(\theta) d\theta}$$

Result: ensemble of algorithms rather than a single one θ_{ML}

- Ensemble usually outperforms single best model
- Posterior capture all dependencies from the training data that the model could extract and may be used as a new prior later

Predictions of Bayesian ML models

Testing stage:

- From training we have a posterior distribution $p\left(\theta \mid X_{tr}, Y_{tr}\right)$
- New data point x arrives
- We need to compute the predictive distribution on its hidden value y

Ensembling w.r.t. posterior over the parameters θ:

$$p\left(y \mid x, X_{tr}, Y_{tr}\right) = \int p(y \mid x, \theta) p\left(\theta \mid X_{tr}, Y_{tr}\right) d\theta$$

Bayesian ML models

Training stage:

$$p\left(\theta \mid X_{tr}, Y_{tr}\right) = \frac{p\left(Y_{tr} \mid X_{tr}, \theta\right) p(\theta)}{\int p\left(Y_{tr} \mid X_{tr}, \theta\right) p(\theta) d\theta}$$

Testing stage:

$$p\left(y \mid x, X_{tr}, Y_{tr}\right) = \int p(y \mid x, \theta) p\left(\theta \mid X_{tr}, Y_{tr}\right) d\theta$$

Bayesian ML models

Training stage:

$$p\left(\theta \mid X_{tr}, Y_{tr}\right) = \frac{p\left(Y_{tr} \mid X_{tr}, \theta\right) p(\theta)}{\boxed{\int p\left(Y_{tr} \mid X_{tr}, \theta\right) p(\theta) d\theta}}$$

May be intractable

Testing stage:

$$p\left(y \mid x, X_{tr}, Y_{tr}\right) = \boxed{\int p(y \mid x, \theta) p\left(\theta \mid X_{tr}, Y_{tr}\right) d\theta}$$

When are the integrals tractable?
What can we do if they are intractable?