Probabilistic Machine Learning
Lecture 26
Making Decisions

Philipp Hennig
20 July 2021
<table>
<thead>
<tr>
<th>#</th>
<th>date</th>
<th>content</th>
<th>Ex</th>
<th>#</th>
<th>date</th>
<th>content</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>20.04.</td>
<td>Introduction</td>
<td>1</td>
<td>14</td>
<td>09.06.</td>
<td>Generalized Linear Models</td>
</tr>
<tr>
<td>2</td>
<td>21.04.</td>
<td>Reasoning under Uncertainty</td>
<td>15</td>
<td>15</td>
<td>15.06.</td>
<td>Exponential Families</td>
</tr>
<tr>
<td>3</td>
<td>27.04.</td>
<td>Continuous Variables</td>
<td>2</td>
<td>16</td>
<td>16.06.</td>
<td>Graphical Models</td>
</tr>
<tr>
<td>4</td>
<td>28.04.</td>
<td>Monte Carlo</td>
<td>17</td>
<td>17</td>
<td>22.06.</td>
<td>Factor Graphs</td>
</tr>
<tr>
<td>5</td>
<td>04.05.</td>
<td>Markov Chain Monte Carlo</td>
<td>3</td>
<td>18</td>
<td>23.06.</td>
<td>The Sum-Product Algorithm</td>
</tr>
<tr>
<td>6</td>
<td>05.05.</td>
<td>Gaussian Distributions</td>
<td>4</td>
<td>19</td>
<td>29.06.</td>
<td>Example: Modelling Topics</td>
</tr>
<tr>
<td>7</td>
<td>11.05.</td>
<td>Parametric Regression</td>
<td>5</td>
<td>20</td>
<td>30.06.</td>
<td>Mixture Models</td>
</tr>
<tr>
<td>8</td>
<td>12.05.</td>
<td>Learning Representations</td>
<td>6</td>
<td>21</td>
<td>06.07.</td>
<td>EM</td>
</tr>
<tr>
<td>9</td>
<td>18.05.</td>
<td>Gaussian Processes</td>
<td>7</td>
<td>22</td>
<td>07.07.</td>
<td>Variational Inference</td>
</tr>
<tr>
<td>10</td>
<td>19.05.</td>
<td>Understanding Kernels</td>
<td>8</td>
<td>23</td>
<td>13.07.</td>
<td>Tuning Inference Algorithms</td>
</tr>
<tr>
<td>12</td>
<td>25.05.</td>
<td>An Example for GP Regression</td>
<td>10</td>
<td>25</td>
<td>20.07.</td>
<td>Outlook</td>
</tr>
<tr>
<td>13</td>
<td>08.06.</td>
<td>GP Classification</td>
<td>11</td>
<td>26</td>
<td>21.07.</td>
<td>Revision</td>
</tr>
<tr>
<td>14</td>
<td>09.06.</td>
<td>Generalized Linear Models</td>
<td></td>
<td>15</td>
<td>15.06.</td>
<td>Exponential Families</td>
</tr>
<tr>
<td>15</td>
<td>15.06.</td>
<td>Exponential Families</td>
<td></td>
<td>16</td>
<td>16.06.</td>
<td>Graphical Models</td>
</tr>
<tr>
<td>16</td>
<td>16.06.</td>
<td>Graphical Models</td>
<td></td>
<td>17</td>
<td>22.06.</td>
<td>Factor Graphs</td>
</tr>
<tr>
<td>17</td>
<td>22.06.</td>
<td>Factor Graphs</td>
<td></td>
<td>18</td>
<td>23.06.</td>
<td>The Sum-Product Algorithm</td>
</tr>
<tr>
<td>18</td>
<td>23.06.</td>
<td>The Sum-Product Algorithm</td>
<td></td>
<td>19</td>
<td>29.06.</td>
<td>Example: Modelling Topics</td>
</tr>
<tr>
<td>19</td>
<td>29.06.</td>
<td>Example: Modelling Topics</td>
<td></td>
<td>20</td>
<td>30.06.</td>
<td>Mixture Models</td>
</tr>
<tr>
<td>20</td>
<td>30.06.</td>
<td>Mixture Models</td>
<td></td>
<td>21</td>
<td>06.07.</td>
<td>EM</td>
</tr>
<tr>
<td>21</td>
<td>06.07.</td>
<td>EM</td>
<td></td>
<td>22</td>
<td>07.07.</td>
<td>Variational Inference</td>
</tr>
<tr>
<td>22</td>
<td>07.07.</td>
<td>Variational Inference</td>
<td></td>
<td>23</td>
<td>13.07.</td>
<td>Tuning Inference Algorithms</td>
</tr>
</tbody>
</table>
The Toolbox

Framework:

\[ \int p(x_1, x_2) \, dx_2 = p(x_1) \quad p(x_1, x_2) = p(x_1 \mid x_2)p(x_2) \quad p(x \mid y) = \frac{p(y \mid x)p(x)}{p(y)} \]

Modelling:

- graphical models
- Gaussian distributions
- (deep) learnt representations
- Kernels
- Markov Chains
- Exponential Families / Conjugate Priors
- Factor Graphs & Message Passing

Computation:

- Monte Carlo
- Linear algebra / Gaussian inference
- maximum likelihood / MAP
- Laplace approximations
- EM / variational approximations
So you’ve got yourself a posterior ...now what?

Taking a decision means *conditioning* on a variable you control.

\[
p(w' \mid \text{run}) \quad p(w' \mid \text{diet})
\]
probabilistic models can provide predictions $p(x \mid a)$ for a variable $x$ \textit{conditional} on an action $a$

given the choice, which value of $a$ do you prefer?
Decision Theory
The limit of probabilistic reasoning?

- probabilistic models can provide predictions $p(x \mid a)$ for a variable $x$ conditional on an action $a$
- given the choice, which value of $a$ do you prefer?

- assign a loss or utility $\ell(x)$
- choose $a$ such that it minimizes expected loss

$$a_\ast = \arg \min_a \int \ell(x)p(x \mid a) \, dx$$
Expected Regret/utility
if you keep having to take the same decision, optimise the sum of its return

- consider *independent* draws $x_i$ with $x_i \sim p(x | a_i)$
- choose all $a_i = a_*$ to minimize the accumulated loss

$$L(n) = \mathbb{E}_p \left[ \sum_i x_i \right]$$

- but what if you *don’t know* $p$?
Motivating (Historical) Example

Experimental Design
Perhaps we shouldn’t rule out an option yet if the posteriors over their expected return overlaps with that of our current guess for the best option?

▶ Assume $K$ choices.
▶ Taking choice $k \in [1, \ldots, K]$ at time $i$ yields binary (Bernoulli) reward/loss $x_i$ with probability $\pi_k \in [0, 1]$, iid.
▶ conjugate priors $p(\pi_k) = B(\pi, a, b) = B(a, b)^{-1}\pi^{a-1}(1-\pi)^{b-1}$
▶ posteriors from $n_k$ trys of choice $k$ with $m_k$ successes:
  $p(\pi_k | n_k, m_k) = B(\pi_k; a + m_k, b + (n_k - m_k))$
▶ for $a, b \to 0$, posterior has mean and variance
  \[ \bar{\pi}_k := \mathbb{E}_p(\pi_k | n_k, m_k)[\pi] = \frac{m_k}{n_k}, \quad \sigma^2_k := \text{var}_p(\pi_k | n_k, m_k)[\pi] = \frac{m_k(n_k - m_k)}{n_k^2(n_k + 1)} = O(n_k^{-1}) \]

Choose option $k$ that maximizes $\bar{\pi}_k + c\sqrt{\sigma^2_k}$ for some $c$. Which $c$?
Perhaps we shouldn’t rule out an option yet if the posteriors over their expected return overlaps with that of our current guess for the best option?

Choose option $k$ that maximizes $\bar{\pi}_k + c\sqrt{\sigma^2_k}$ for some $c$. Which $c$?

- A large $c$ ensures uncertain options are preferred. If we make it too large, we will only explore.
- A small $c$ largely ignores uncertainty. We will only exploit.
- Idea: Let $c$ grow slowly over time, at rate less than $O(n_k^{1/2})$. Then variance of chosen options will drop faster than $c$ grows, so their exploration will stop, unless their mean is good. But unexplored choices will eventually become dominant, thus always explored eventually.
Not just for Bernoulli variables!
posterior contraction rates are universal

**Theorem (Chernoff-Hoeffding)**

Let $X_1, \ldots, X_n$ be random variables with common range $[0, 1]$ and such that $\mathbb{E}[X_t | X_1, \ldots, X_{t-1}] = \mu$. Let $S_n = X_1 + \cdots + X_n$. Then for all $a \geq 0$,

\[
p(S_n - n\mu \leq -a) \leq e^{-2a^2/n} \quad \text{and} \quad p(S_n - n\mu \geq a) \leq e^{-2a^2/n}
\]
The Multi-Armed Bandit Setting
Discrete-Choice Experimental Design

Definitions:

- A **K-armed bandit** is a collection $X_{kn}$ of random variables, $1 \leq k \leq K$, $n \geq 1$ where $k$ is the arm of the bandit. Successive plays of $k$ yield rewards $X_{k1}, X_{k2}, \ldots$ which are **independent and identically distributed** according to an unknown $p$ with $\mathbb{E}_p(X_{ki}) = \mu_i$.

- A **policy** $A$ chooses the next machine to play at time $n$, based on past plays and rewards.

- Let $T_k(n)$ be number of times machine $k$ was played by $A$ during the first $n$ plays. The **regret** of $A$ is

\[
R_A(n) = \mu^* \cdot n - \sum_j \mu_j \cdot \mathbb{E}_p[T_j(n)] \quad \text{with } \mu^* := \max_{1 \leq k \leq K} \mu_k
\]
Algorithm: Let $\bar{x}_j$: empirical average of rewards from $j$, $n_j$: number of plays at $j$ in $n$ plays

1. \textbf{procedure} UCB(K) \\
2. \hspace{1em} play each machine once \\
3. \hspace{1em} \textbf{while} true \textbf{ do} \\
4. \hspace{2em} play $j = \arg \max \left( \bar{x}_j + \sqrt{\frac{2 \log n}{n_j}} \right)$ \\
5. \hspace{1em} \textbf{end while} \\
6. \textbf{end procedure}
### Theorem (Auer, Cesa-Bianchi, Fischer)

Consider $K$ machines ($K > 1$) having arbitrary reward distributions $P_1, \ldots, P_K$ with support in $[0, 1]$ and expected values $\mu_i = \mathbb{E}_P(X_i)$. Let $\Delta_i := \mu_* - \mu_i$. Then, the expected regret of UCB after any number $n$ of plays is at most

$$\mathbb{E}_P[R_A(n)] \leq \left[ 8 \sum_{i: \mu_i \leq \mu_*} \left( \frac{\log n}{\Delta_i} \right) \right] + \left( 1 + \frac{\pi^2}{3} \right) \left( \sum_j \Delta_j \right)$$

Nb: The sums are over $K$, not $n$. So the regret is $O(K \log n)$. UCB plays a sub-optimal arm at most logarithmically often.
$K = 3$, binary rewards

$$\sum_t n_t \quad p = 50\%$$

$$\sum_t n_t \quad p = 55\%$$

$$\sum_t n_t \quad p = 45\%$$

$N \quad \text{regret bound} \\ N \quad \text{expected regret} \\ N \quad \text{sampled regret}$
Multi-Armed Bandit Algorithms

- apply to independent, discrete choice problems with stochastic pay-off
- algorithms based on upper confidence bounds incur regret bounded by $O(\log n)$
- this even applies for the adversarial setting (Auer, Cesa-Bianchi, Freund, Schapire, 1995)
Multi-Armed Bandit Algorithms

- apply to independent, discrete choice problems with stochastic pay-off
- algorithms based on upper confidence bounds incur regret bounded by $O(\log n)$
- this even applies for the adversarial setting (Auer, Cesa-Bianchi, Freund, Schapire, 1995)

Unfortunately...

- No problem is ever discrete, finite and independent
- in a continuous problem, no “arm” can and should ever be played twice
- in many prototyping settings, early exploration is free
Continuous-Armed Bandits

example application: parameter optimization

\[ p(y \mid x) = \mathcal{N}(y; f(x), \sigma^2) \]

\[ x_\star = \arg\min_{x \in \mathcal{X}} f(x) = ? \]

\[ R(T) := \sum_{t=1}^{T} f(x_t) - f(x_\star) \]
Continuous-Armed Bandits

example application: parameter optimization

\[ p(y | x) = \mathcal{N}(y; f_x, \sigma^2) \quad p(f) = \mathcal{GP}(f; \mu, k) \quad \Rightarrow \quad p_{\text{min}}(x^* = x) = \int_{\mathbb{R}} \int_{\mathcal{D}} \mathbb{I}(f(x) < f(\tilde{x})) \, d\tilde{x} \, dp(f | y) \]
GP Upper Confidence Bound
Evaluate optimistically, where the function may be low

utility under $p(f \mid y) = \mathcal{GP}(f; \mu_{t-1}, \sigma^2_t)$

$$u_i(x) = \mu_{i-1}(x) - \sqrt{\beta_t \sigma_{t-1}(x)}$$

choose $x_t$ as $x_t = \arg\min_{x \in \mathcal{D}} u(x)$

Theorem (Srinivas et al., 2009)
Let $\delta \in (0, 1)$ and $\beta_t = 2 \log(|\mathcal{D}| t^2 \pi^2 / 6\delta)$. Running GP-UCB with $\beta_t$ for a sample $f \sim \mathcal{GP}(\mu, k)$,

$$p\left(R_T \leq \sqrt{8T \beta_T \gamma_T / \log(1 + \sigma^2) \quad \forall T \geq 1}\right) \geq 1 - \delta$$

thus $\lim_{T \to \infty} R_T / T = 0$ ("no regret").
Evaluate optimistically, where the function may be low.

utility under $p(f \mid y) = \mathcal{G}\mathcal{P}(f; \mu_{t-1}, \sigma_{t-1}^2)$

$u_i(x) = \mu_{i-1}(x) - \sqrt{\beta_t \sigma_{t-1}(x)}$

choose $x_t$ as $x_t = \arg\min_{x \in \mathbb{D}} u(x)$

Theorem (Srinivas et al., 2009)

Assume that $f \in \mathcal{H}_k$ with $\|f\|_k^2 \leq B$, and the noise is zero-mean and $\sigma$-bounded almost surely. Let $\delta \in (0, 1)$ and $\beta_t = 2B + 300\gamma_t \log^3(t/\delta)$. Running GP-UCB with $\beta_t$ and $p(f) = \mathcal{G}\mathcal{P}(f; 0, k)$,

$$p \left( R_T \leq \sqrt{8T\beta_T \gamma_T / \log(1 + \sigma^2)} \quad \forall T \geq 1 \right) \geq 1 - \delta$$

thus $\lim_{T \to \infty} R_T / T = 0$ ("no regret").
What if you have budget for several experiments?
Entropy Search

evaluate where you expect to learn most about the minimum

[Villemonteix et al., 2009; Hennig & Schuler, 2012]

\[ p(f) = \mathcal{GP}(f; m, k) \]
\[ p(y | f) = \mathcal{N}(y; f_x, \sigma^2) \]
\[ p(f | y) = \mathcal{N}(f; \mu, k) \]

\[ \bar{\mu}_a = \mu_a + \kappa_{a*}^{-1} (y_* - \mu_*) \]
\[ = \mu_a + \kappa_{a*}^{-1/2} \cdot \kappa_{**}^{-1/2} (y_* - \mu_*) \]
\[ = : L_{a*} \quad u \sim \mathcal{N}(0, I) \]
\[ \bar{\kappa}_{ab} = \kappa_{ab} - \kappa_{a*} \kappa_{**}^{-1} \kappa_{*b} \]
\[ = \kappa_{ab} - L_{a*}L_{*b} \]

use this to predict \( \hat{p}_{\min}(x) \) under \( p(f | y, y_{t+1}) \)
(requires nontrivial numerics)
Entropy Search

evaluate where you expect to learn most about the minimum

\[ p(f) = \mathcal{GP}(f; m, k) \] and
\[ p(y | f) = \mathcal{N}(y; f_x, \sigma^2) \]
gives
\[ p(f | y) = \mathcal{N}(f; \mu, k), \]
and
\[ \mu_a = \mu_a + \kappa_{a*}^{-1}(y_* - \mu_*) \]
\[ = \mu_a + \kappa_{a*}^{-1/2} \cdot \kappa_*^{-1/2}(y_* - \mu_*) \]
\[ =: L_{a*} \]
\[ u \sim \mathcal{N}(0, I) \]
\[ \kappa_{ab} = \kappa_{ab} - \kappa_{a*} \kappa_*^{-1} \kappa_{b*} \]
\[ = \kappa_{ab} - L_{a*} L_{b*} \]

\[ \mu_a = \mu_a + \kappa_{a*}^{-1}(y_* - \mu_*) \]
\[ \kappa_{ab} = \kappa_{ab} - \kappa_{a*} \kappa_*^{-1} \kappa_{b*} \]

\[ \hat{p}_{\min}(x) \] under \[ p(f | y, y_{t+1}) \]

requires nontrivial numerics
Entropy Search
evaluate where you expect to learn most about the minimum

\[ p(f) = \mathcal{GP}(f; m, k) \] and
\[ p(y | f) = \mathcal{N}(y; f_x, \sigma^2) \]
gives
\[ p(f | y) = \mathcal{N}(f; \mu, k), \] and
\[ \bar{\mu}_a = \mu_a + \kappa_{a*}^{-1} (y_* - \mu_*) \]
\[ = \mu_a + \kappa_{a*}^{-1/2} \cdot \kappa_{**}^{-1/2} (y_* - \mu_*) \]
\[ = : L_{a*} \sim \mathcal{N}(0, I) \]
\[ \bar{\kappa}_{ab} = \kappa_{ab} - \kappa_{a*} \kappa_{**}^{-1} \kappa_{*b} \]
\[ = \kappa_{ab} - L_{a*} L_{*b} \]

use this to predict \( \hat{p}_{\min}(x) \) under \( p(f | y, y_{t+1}) \)
(requires nontrivial numerics)
Entropy Search

evaluate where you expect to learn most about the minimum

[Villemonteix et al., 2009; Hennig & Schuler, 2012]

\[ p(f) = \mathcal{GP}(f; m, k) \] and
\[ p(y | f) = \mathcal{N}(y; f_x, \sigma^2) \]
gives
\[ p(f | y) = \mathcal{N}(f; \mu, k), \]
and
\[ \bar{\mu}_a = \mu_a + \kappa_a \kappa_*^{-1}(y_* - \mu_*) \]
\[ = \mu_a + \kappa_a \kappa_*^{-1/2} \cdot \frac{\kappa_*^{-1/2}(y_* - \mu_*)}{u \sim \mathcal{N}(0, I)} \]
\[ \bar{\kappa}_{ab} = \kappa_{ab} - \kappa_a \kappa_*^{-1} \kappa_{*b} \]
\[ = \kappa_{ab} - L_a L_{*b} \]

▶ use this to predict \( \hat{p}_{\text{min}}(x) \) under \( p(f \mid y, y_{t+1}) \)

(requires nontrivial numerics)
Entropy Search
evaluate where you expect to learn most about the minimum

\[ p(f) = \mathcal{GP}(f; m, k) \]
\[ p(y \mid f) = \mathcal{N}(y; f_x, \sigma^2) \]
gives
\[ p(f \mid y) = \mathcal{N}(f; \mu, k), \]
and
\[ \bar{\mu}_a = \mu_a + \kappa_a \kappa^{-1}(y_\ast - \mu_\ast) \]
\[ = \mu_a + \kappa_a \kappa^{-1/2} \cdot \kappa^{-1/2}(y_\ast - \mu_\ast) \]
\[ = \mu_a + L_{a*} \cdot (y_\ast - \mu_\ast) \]
\[ u \sim \mathcal{N}(0, I) \]
\[ \bar{\kappa}_{ab} = \kappa_{ab} - \kappa_a \kappa_{**} \kappa_{*b} \]
\[ = \kappa_{ab} - L_{a*} L_{*b} \]

use this to predict \( \hat{p}_{\min}(x) \) under \( p(f \mid y, y_{t+1}) \)
(requires nontrivial numerics)
Entropy Search
evaluate where you expect to learn most about the minimum

[Villemonteix et al., 2009; Hennig & Schuler, 2012]

Don't evaluate where you think the minimum lies!
Instead, evaluate where you expect to learn most about the minimum!

\[ \mathbb{H}(p) := -\int p(x) \log \frac{p(x)}{b(x)} \, dx \]

with base measure \( b \). Use utility

\[ u(x) = \mathbb{H}_t(p_{\text{min}}) - \mathbb{E}_{y_{t+1}}[\mathbb{H}_{t+1}(p_{\text{min}})] \]
Information vs. Regret

Entropy Search is qualitatively different from regret-based formulations

Settings in which information-based search is preferrable

▶ “prototyping-phase” followed by “product release”
▶ structured uncertainty with variable signal-to-noise ratio
▶ “multi-fidelity”: Several experimental channels of different cost and quality, e.g.
  ▶ simulations vs. physical experiments
  ▶ training a learning model for a variable time
  ▶ using variable-size datasets

Regret-based optimization is easy to implement and works well on standard problems. But it is a strong simplification of reality, in which many practical complications can not be phrased.
Bayesian Optimization in Practice
recent (and not so recent) libraries

- https://amzn.github.io/emukit/
- https://github.com/HIPS/Spearmint
- https://github.com/hyperopt
- https://github.com/automl
- https://sigopt.com/product/
Summary — Experimental Design

- The **bandit setting** formalizes iid. sequential decision making under uncertainty.
- Bandit algorithms can achieve “no regret” performance, even without explicit probabilistic priors.
- **Bayesian optimization** extends to continuous domain.
- It lies right at the intersection of computational and physical learning.
- Requires significant computational resources to run a numerical optimizer inside the loop.
- Allows rich formulation of global, stochastic, continuous, structured, multi-channel design problems.
- Is currently the state of the art in the solution of challenging optimization problems.