Bayesian optimization, bandits and Thompson sampling

Nando de Freitas
February 2013
Multi-armed bandit problem
Multi-armed bandit problem

- Actions \( \{1, 2, \ldots, K\} \)
- Reward(s) \( x(t) \in [0, 1]^K \)
- Sequence of trials \( t = 1, 2, \ldots, T \)

- Trade-off between Exploration and Exploitation
- Regret = Player reward – Reward of best action
Bayesian optimization

1: for $t = 1, 2, \ldots$ do
2: \hspace{1em} Find $x_t$ by combining attributes of the posterior distribution in a utility function $u$ and maximizing:
   \hspace{1em} $x_t = \text{argmax}_x u(x | \mathcal{D}_{1:t-1})$.
3: \hspace{1em} Sample the objective function:
   \hspace{1em} $y_t = f(x_t) + \varepsilon_t$.
4: \hspace{1em} Augment the data $\mathcal{D}_{1:t} = \mathcal{D}_{1:t-1}, (x_t, y_t)$ and update the GP.
5: end for
Exploration-exploitation tradeoff

Recall the expressions for GP prediction:

\[
P(y_{t+1} | D_{1:t}, x_{t+1}) = \mathcal{N}(\mu_t(x_{t+1}), \sigma_t^2(x_{t+1}) + \sigma_{\text{noise}}^2)
\]

\[
\mu_t(x_{t+1}) = k^T[K + \sigma_{\text{noise}}^2 I]^{-1}y_{1:t}
\]

\[
\sigma_t^2(x_{t+1}) = k(x_{t+1}, x_{t+1}) - k^T[K + \sigma_{\text{noise}}^2 I]^{-1}k
\]

We should choose the next point \( x \) where the mean is high (exploitation) and the variance is high (exploration).

We could balance this tradeoff with an acquisition function as follows:

\[
\mu(x) + \kappa \sigma(x)
\]
Acquisition functions

- *aka* infill, figure of merit

- acquisition function guides the optimization by determining which $x_{t+1}$ to observe next.

- uses predictive posterior to combine exploration (high-variance regions) and exploitation (high-mean regions).

- optimize to find sample point (can be done cheaply/approximately).
An acquisition function: Probability of Improvement

$$\text{PI}(x) = P(f(x) \geq \mu^+ + \xi)$$

$$= \Phi \left( \frac{\mu(x) - \mu^+ - \xi}{\sigma(x)} \right)$$

$\eta^+$ best observed value

$\eta^+ = \mu^+ + \xi$ small
People as Bayesian reasoners
Bayes and decision theory

Utilitarian view: We need models to make the right decisions under uncertainty. Inference and decision making are intertwined.

Learned posterior

\[
\begin{aligned}
P(x=\text{healthy}|\text{data}) &= 0.9 \\
P(x=\text{cancer}|\text{data}) &= 0.1
\end{aligned}
\]

Cost/Reward model \( u(x,a) \)

<table>
<thead>
<tr>
<th></th>
<th>( a = \text{no treatment} )</th>
<th>( a = \text{treatment} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( x = \text{healthy} )</td>
<td>0</td>
<td>-30</td>
</tr>
<tr>
<td>( x = \text{cancer} )</td>
<td>-100</td>
<td>-20</td>
</tr>
</tbody>
</table>

We choose the action that maximizes the expected utility, or equivalently, which minimizes the expected cost.

\[
EU(a) = \sum_x u(x,a) P(x|\text{data})
\]

\[
\begin{aligned}
EU(a=\text{treatment}) &= u(\text{healthy}, \text{treatment}) P(x=\text{healthy}|\text{data}) + u(\text{cancer}, \text{treated}) P(x=\text{cancer}|\text{data}) \\
&= (-30)(0.9) + (-20)(0.1) \\
EU(a=\text{no treatment}) &= \end{aligned}
\]
An expected utility criterion

At iteration $n+1$, choose the point that minimizes the distance to the objective evaluated at the maximum $x^*$:

$$
x_{n+1} = \arg \min_x \mathbb{E}(\|f_{n+1}(x) - f(x^*)\| | \mathcal{D}_n)
$$

$$
= \arg \min_x \int \|f_{n+1}(x) - f(x^*)\| p(f_{n+1} | \mathcal{D}_n) df_{n+1}
$$

We don’t know the true objective at the maximum. To overcome this, Mockus proposed the following acquisition function:

$$
x = \arg \max_x \mathbb{E}(\max\{0, f_{n+1}(x) - f_{\text{max}}\} | \mathcal{D}_n)
$$
Expected improvement

\[ x = \arg \max_x \mathbb{E}(\max\{0, f_{n+1}(x) - f_{\max}^n\} \mid \mathcal{D}_n) \]

For this acquisition, we can obtain an analytical expression:

\[
EI(x) = \begin{cases} 
(\mu(x) - \mu^+ - \xi) \Phi(Z) + \sigma(x) \phi(Z) & \text{if } \sigma(x) > 0 \\
0 & \text{if } \sigma(x) = 0
\end{cases}
\]

\[ Z = \frac{\mu(x) - \mu^+ - \xi}{\sigma(x)} \]

where \( \phi(\cdot) \) and \( \Phi(\cdot) \) denote the PDF and CDF of the standard Normal
A third criterion: GP-UCB

Define the regret and cumulative regret as follows:

$$r(x) = f(x^*) - f(x)$$

$$R_T = r(x_1) + \cdots + r(x_T)$$

The GP-UCB criterion is as follows:

$$\text{GP-UCB}(x) = \mu(x) + \sqrt{\nu \beta_t \sigma(x)}$$

Beta is set using a simple concentration bound:

With $\nu = 1$ and $\beta_t = 2 \log(t^{d/2} + t^{-2/3}/\delta)$, it can be shown with high probability that this method is no regret, i.e. $\lim_{T \to \infty} R_T/T = 0$. This in turn implies a lower-bound on the convergence rate for the optimization problem.

[Srinivas et al, 2010]
A fourth criterion: Thompson sampling

$X_{t+1}$
next point to try
Acquisition functions

\[ \mu^+ = \text{argmax}_{x_i \in x_{1:t}} \mu(x_i) \]

- Probability of Improvement

\[
\text{PI}(x) = \Phi \left( \frac{\mu(x) - \mu^+ - \xi}{\sigma(x)} \right)
\]

Kushner 1964

- Expected Improvement

\[
\text{EI}(x) = (\mu(x) - \mu^+ - \xi)\Phi(Z) + \sigma(x)\phi(Z)
\]

\[
Z = \frac{\mu(x) - \mu^+ - \xi}{\sigma(x)}
\]

Mockus 1978

- Upper Confidence Bound

\[
\text{GP-UCB}(x) = \mu(x) + \sqrt{\nu \tau_t \sigma(x)}
\]

Srinivas et al. 2010

\[
\text{PI} \quad \xi=0.01, \xi=0.10, \xi=1.00
\]

\[
\text{EI} \quad \xi=0.01, \xi=0.10, \xi=1.00
\]

\[
\text{GP-UCB} \quad \nu=0.2, \nu=1.0, \nu=2.0
\]
Portfolios of acquisition functions help
Why Bayesian Optimization works
Intelligent user interfaces
Example: Tuning NP hard problem solvers
Why random tuning works sometimes
Example: Tuning random forests
Example: Tuning hybrid Monte Carlo

Table 4.1: Mean squared test error for the robot arm data set.

<table>
<thead>
<tr>
<th>Method</th>
<th>Mean Squared Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rios Insua and Muller’s (1998) MLP with reversible-jump MCMC</td>
<td>0.00620</td>
</tr>
<tr>
<td>Mackay’s (1992) Gaussian approximation with highest evidence</td>
<td>0.00573</td>
</tr>
<tr>
<td>Neal’s (1996) HMC</td>
<td>0.00554</td>
</tr>
<tr>
<td>Neal’s (1996) HMC with ARD</td>
<td>0.00549</td>
</tr>
<tr>
<td>Reversible-jump MCMC with Bayesian model by Andrieu et al.</td>
<td>0.00502</td>
</tr>
<tr>
<td>Adaptive HMC (Median Error)</td>
<td>0.00499</td>
</tr>
<tr>
<td>Adaptive HMC (Mean Error)</td>
<td>0.00498 ± 0.00012</td>
</tr>
</tbody>
</table>

Table 4.2: Classification error on the validation set of the Dexter data set.

<table>
<thead>
<tr>
<th>Method</th>
<th>Classification Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>New-Bayes-mm-sel</td>
<td>0.0800</td>
</tr>
<tr>
<td>Adaptive HMC (Mean error)</td>
<td>0.0730 ± 0.0096</td>
</tr>
<tr>
<td>Adaptive HMC (Median error)</td>
<td>0.0700</td>
</tr>
<tr>
<td>Adaptive HMC + Majority Voting</td>
<td>0.0667</td>
</tr>
</tbody>
</table>
The games industry, rich in sophisticated large-scale simulators, is a great environment for the design and study of automatic decision making systems.
Hierarchical policy example

- **High-level** model-based learning for deciding when to navigate, park, pickup and dropoff passengers.

- **Mid-level** active path learning for navigating a topological map.

- **Low-level** active policy optimizer to learn control of continuous non-linear vehicle dynamics.
Active Path Finding in Middle Level

- Mid-level *Navigate* policy generates sequence of waypoints on a topological map to navigate from a location to a destination. $V(\theta)$ value function represents the path length from the current node, to the target.
Low-Level: Trajectory following

TORCS: 3D game engine that implements complex vehicle dynamics complete with manual and automatic transmission, engine, clutch, tire, and suspension models.
Hierarchical systems apply to many robot tasks – key to build large systems

We used TORCS: A 3D game engine that implements complex vehicle dynamics complete with manual and automatic transmission, engine, clutch, tire, and suspension models.
Gaze planning

Digits Experiment:

Face Experiment:
Next lecture

In the next lecture, we embark on our quest to learn all about random forests. We will begin by learning about decision trees.