Bayesian Estimation Example Using PyMC

SciPy 2010 Lightning Talk

Dan Williams
Life Technologies
Austin TX
What is PyMC?

PyMC is a Python module that provides tools for Bayesian analysis.

• NOTE: I am not a contributer to this project--just an enthusiastic user!
Motivation

• Suppose we have a series of short DNA sequences, each known to cause one of two experimental outcomes:

<table>
<thead>
<tr>
<th>Sequence</th>
<th>Outcome</th>
</tr>
</thead>
<tbody>
<tr>
<td>CGTCGGAGGTACATGATTGGAAAGAAACCT</td>
<td>Y</td>
</tr>
<tr>
<td>GCGCCTTTTGCACTCTCTTAATCTCAGTCA</td>
<td>X</td>
</tr>
<tr>
<td>TTAAATAGCAGAGACACTTCTACTGATAC</td>
<td>Y</td>
</tr>
<tr>
<td>CCAAGAGCCTCGTAATTAAGTGATTGCAATA</td>
<td>Y</td>
</tr>
<tr>
<td>TTATGACGTCGTTTCGAGTGATTTGTCTT</td>
<td>X</td>
</tr>
<tr>
<td>……</td>
<td>……</td>
</tr>
</tbody>
</table>

• We want to train a statistical model to predict the outcome from any arbitrary sequence.
Motivation (continued)

- A common strategy looks for motifs in the sequences and correlates them to outcomes.
  
  - Simple example: Nucleotide “A” may follow nucleotide “T” in the sequences more frequently for outcome X than for outcome Y,

  \[ P(A | T, X) > P(A | T, Y) \]

- If you know such probabilities, you can create a variety of scoring models for arbitrary input sequences to help predict experiment outcome.
But how do we get the probabilities?

• Option #1 - Maximum Likelihood Method (Frequentist Approach)
  - Derive probabilities from a large experimental set with measured outcomes.

• Option #2 - Maximum a Posteriori (MAP) Estimation (Bayesian Approach)
  - Use Bayes’ theorem to combine researcher intuition with a small experimental dataset to estimate probabilities.
  - *PyMC makes this easy!*
Python Bayesian Estimation Workflow

• Start with Bayes’ theorem:

\[ P(\theta | D) = \frac{P(D | \theta) \cdot P(\theta)}{P(D)} \]

\[ D = \text{observed data} \]
\[ \theta = \text{scoring model parameters} \]
Python Bayesian Estimation Workflow

• Specify the prior distribution:

```python
import numpy as np
from pymc import Dirichlet  # conjugate prior
alpha = np.array([30.0, 25.0, 20.0, 25.0])
prob_dist = Dirichlet('prob_dist', alpha)
```

\[
P(\theta | D) = \frac{P(D | \theta) \cdot P(\theta)}{P(D)}
\]
Python Bayesian Estimation Workflow

- Specify the experimental data:
  
  ```python
  exp_data = np.array([1, 1, 3, 2, 2, 1, 0, ...])
  ```

\[
P(\theta | D) = \frac{P(D | \theta) \cdot P(\theta)}{P(D)}
\]
Python Bayesian Estimation Workflow

• Specify the value to maximize using numerical simulation, as well as the expected form of the posterior distribution:

   from pymc import Categorical
   f_x = Categorical('cat', prob_dist, value=exp_data, observed=True)

\[
P(\theta \mid D) = \frac{P(D \mid \theta) \cdot P(\theta)}{P(D)}
\]
Python Bayesian Estimation Workflow

- Compute maximum \textit{a posteriori} estimates of the probabilities:

```python
from pymc import MAP, Model
model = Model({'f_x' : f_x, 'prob_dist' : prob_dist})
M = MAP(model)
M.fit()  # Nelder-Mead Optimization
```

- The MAP estimates are now contained in the \texttt{M.prob_dist} value:

```python
>>> print M.prob_dist.value
[ 0.19472259  0.26842748  0.25265728]
```

\[
P(\theta | D) = \frac{P(D | \theta) \cdot P(\theta)}{P(D)}
\]
Testing Set Results: A Predictive Model Parameterized by Informed Priors vs. the Same Model Parameterized by MAP Estimates

*Predictive Model Built Using Informed Priors Only*

*Predictive Model Built Using MAP Estimation*

Test Set, p=2.0e-15 (Mann-Whitney)

Test Set, p=1.7e-17 (Mann-Whitney)
Thank you!