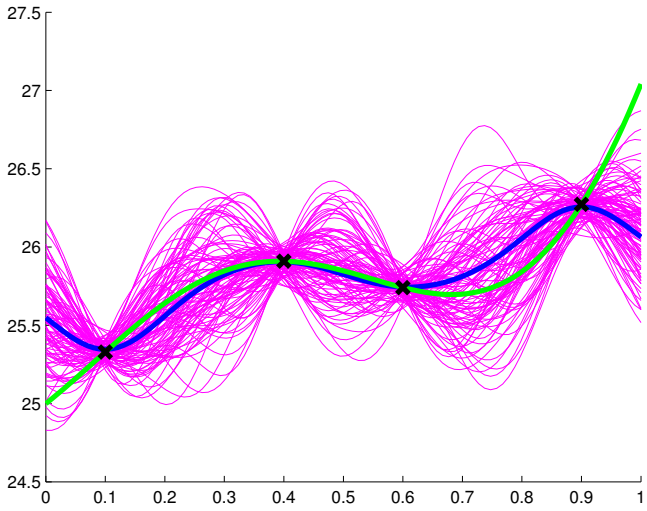
10-715 Advanced Introduction to Machine Learning

Homework 4 Solutions *November 23, 2014*

1 Gaussian Processes (Samy)

1.1 GP Implementation

Our outputs for the toy problem,



Our output for the CCPP dataset,

```
Error: 4.7381

ans =

  469.6517
  457.8241
  457.3179
  436.5464
  437.3415


ans =

    0.2090   -0.0000   -0.0000    0.0000   -0.0000
   -0.0000    1.2859   -0.0000   -0.0000    0.0000
   -0.0000   -0.0000    0.6351    0.0000   -0.0000
    0.0000   -0.0000    0.0000    0.2055    0.0000
   -0.0000    0.0000   -0.0000    0.0000    0.8464
```

This is our implementation.

```
function K = GaussKernel(X1, X2, sigma, h)
  D = dist2(X1, X2);
  K = sigma * exp( -D/(2*h^2) ) ;
end


function [postMean, postVar] = GPRegression(XTrain, YTrain, XTest, hyperParams)

  % prelims
  h = hyperParams.h;
  sigma = hyperParams.sigma;
  L = hyperParams.L;
  alpha = hyperParams.alpha;
  eta = hyperParams.eta;
  priorMean = hyperParams.priorMean;

  % Obtain the predictions
  y = YTrain - priorMean;
  Ktrte = GaussKernel(XTrain, XTest, sigma, h);
  Ktete = GaussKernel(XTest, XTest, sigma, h);
  Ktete(1:5, 1:5),
  v = L \ Ktrte;
  postMean = priorMean + Ktrte' * alpha;
  postVar = Ktete - v' * v;

end


function [hyperParams] = chooseHyperParams(X, Y)
% Picks the best scale and bandwidth by maximizing the marginal likelihood

  % Determine candidates for h, sigma
  numCands = 10;
  hCands = logspace(-1, 1, numCands)' * norm(std(X));
  sigmaCands = logspace(-1, 1, numCands)' * std(Y);
  a1 = repmat(hCands, numCands, 1);
  a2 = repmat(sigmaCands, 1, numCands); a2 = a2(:);
  cands = [a1 a2];

  % set the mean value
  priorMean = mean(Y);
  hyperParams.priorMean = priorMean;
  y = Y - priorMean;
  % set the noise level
  eta = 0.01 * std(Y);
  hyperParams.eta = eta;

  % Now determine the best pair of candidates
  bestNlml = -inf;
  vals = zeros(size(cands,1), 1);
  for i = 1:size(cands, 1)
    h = cands(i, 1);
    sigma = cands(i, 2);
```

```
    [nlml, L, alpha] = normalizedMargLikelihood(h, sigma, X, y, eta);
    vals(i) = nlml;
    if nlml > bestNlml
      bestNlml = nlml;
      hyperParams.h = h;
      hyperParams.sigma = sigma;
      hyperParams.L = L;
      hyperParams.alpha = alpha;
    end
  end

end

function [nlml, L, alpha] = normalizedMargLikelihood(h, sigma, X, y, eta)
  n = size(X, 1);
  K = GaussKernel(X, X, sigma, h) + eta * eye(n);
  L = stableCholesky(K);
  alpha = L' \ (L \ y);
  nlml = -1/2 * y' * alpha - sum(log(diag(L))) - n/2 * log(2*pi);
end


function samples = GPDrawSamples(mu, K, numSamples)
% A function for drawing samples from a Gaussian Process
% V is a num_samples x m matrix, each row corresponding to a sample

  num_pts = size(mu,1);

  L = stableCholesky(K);
  Z = randn(num_pts, numSamples);
  samples = real(bsxfun(@plus, L*Z, mu))';

end
```

1.2 Derivatives using GPs

1. Using the definitions we can write

$$\begin{pmatrix} \mathbf{y} \\ \mathbf{g}_* \end{pmatrix} = \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} \mathbf{K} & \mathbf{J} \\ \mathbf{J}^\top & \mathbf{B} \end{bmatrix}\right)$$

2. When we condition on $\mathbf{y}$ we get,

$$\mathbf{g}_* | \mathbf{X}, \mathbf{y}, x_* \sim \mathcal{N}(\mathbf{J}^\top \mathbf{K}^{-1}\mathbf{y}, \mathbf{B} - \mathbf{J}^\top \mathbf{K}^{-1}\mathbf{J})$$

3. The posterior mean for $y(x_*)$ is $\hat{f}(x_*) = K(x_*, X)\mathbf{K}^{-1}\mathbf{y}$ where $K(x_*, X) \in \mathbb{R}^{1\times n}$, $K(x_*, X)_i = K(x_*, x_i)$.

$$\nabla_{x_*} \hat{f}(x_*) = \mathbf{J}^\top \mathbf{K}^{-1}\mathbf{y}$$

4. Let $Y(x) = \int_0^x y(x)\mathrm{d}x$. Then $\int_a^b y(x) = Y(b) - Y(a)$. If we treat Y as a GP with kernel K_Y, then y and Y are can also be described by Gaussians as given in the previous parts. The covariance of y

would be K_y and that between y and Y would be K_{yY} where,

$$K_y(x_1, x_2) = \frac{\partial^2 K_Y(x_1, x_2)}{\partial x_1 \partial x_2}$$
$$K_{yY}(x_1, x_2) = \frac{\partial K_Y(x_1, x_2)}{\partial x_1}$$

Denoting $\mathbf{Y} = [Y(b), Y(a)]^\top$, we can write the prior over $[\mathbf{y}, \mathbf{Y}]$ as,

$$\begin{pmatrix} \mathbf{y} \\ \mathbf{Y} \end{pmatrix} = \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} \mathbf{K} & \mathbf{P} \\ \mathbf{P}^\top & \mathbf{C} \end{bmatrix}\right)$$

where $\mathbf{K} \in \mathbb{R}^{n \times n}$ is the kernel matrix. $\mathbf{P} \in \mathbb{R}^{n \times 2}$ where $\mathbf{P}_{i1} = K_{yY}(x_i, b)$, $\mathbf{P}_{i2} = K_{yY}(x_i, a)$. And

$$\mathbf{C} = \begin{bmatrix} K_Y(b,b) & K_Y(b,a) \\ K_Y(a,b) & K_Y(a,a) \end{bmatrix}$$

This gives the posterior,

$$\mathbf{Y}|\mathbf{X}, \mathbf{y}, a, b \sim \mathcal{N}(\mathbf{P}^\top \mathbf{K}^{-1} \mathbf{y}, \mathbf{C} - \mathbf{P}^\top \mathbf{K}^{-1} \mathbf{P}) \quad \equiv \mathcal{N}(\xi, \Sigma) \quad \text{(say)}.$$

Let $\mathbf{e} = [1, -1]^\top$. Our estimate of the integral is $\mathbf{Y}^\top \mathbf{e}$. Therefore, the posterior for the integral is $\mathcal{N}(\xi^\top \mathbf{e}, \mathbf{e}^\top \Sigma \mathbf{e}) = \mathcal{N}(\xi_1 - \xi_2, \Sigma_{11} + \Sigma_{22} - 2\Sigma_{12})$.

2 Dirichlet Process Mixture Model(Veeru)

2.1 Understanding the generative process

You should see more clusters in both the problems (2) and (3) compared to case I.

2.2 Inference with Gibbs sampling

2.2.1 Conditional distributions

$$\begin{aligned}
p(z_i = k | x, z_{-i}, \mu) &= \frac{p(z_i = k, x, z_{-i}, \mu)}{\sum_{k'} p(z_i = k', x, z_{-i}, \mu)} \\
&= \frac{p(x | z_i = k, z_{-i}, \mu) p(z_i = k | z_{-i}) p(z_{-i}) p(\mu)}{\sum_{k'} p(x | z_i = k', z_{-i}, \mu) p(z_i = k' | z_{-i}) p(z_{-i}) p(\mu)} \\
&= \frac{p(x | z_i = k, z_{-i}, \mu) p(z_i = k | z_{-i})}{\sum_{k'} p(x | z_i = k', z_{-i}, \mu) p(z_i = k' | z_{-i})} \\
&\propto p(x | z_i = k, z_{-i}, \mu) p(z_i = k | z_{-i}) \\
&\propto p(x_i | z_i = k, \mu_k) p(z_i = k | z_{-i}) \\
&= p(x_i | z_i = k, \mu_k) \frac{n_{-i,k}}{n - 1 + \alpha}
\end{aligned}$$

For the case where $k = K_{new}$, we use a part of the previous derivation.

$$\begin{aligned}
p(z_i = K_{new} | x, z_{-i}, \mu) &\propto p(z_i = K_{new} | z_{-i}) p(x | z_i = K_{new}, z_{-i}, \mu) \\
&\propto p(z_i = K_{new} | z_{-i}) p(x_i | z_i = K_{new}) \\
&= p(z_i = K_{new} | z_{-i}) \int p(x_i | \nu) \pi(\nu) d\nu
\end{aligned}$$

In the last but one step, as $z_i = K_{new}$, the distribution of x_i does not depend on μ. In the last step, we are just considering all possible ν's that can be the cluster centers of the new cluster. Looking at the derivation, it is not difficult to see that the proportionality constant C is same for the both the cases.

2.2.2 Implementation

For k in $\{z_j | j \neq i\}$,

$$p(z_i = k | x, z_{-i}, \mu) = C \frac{n_{-i,k}}{n - 1 + \alpha} f(x_i | \mu_k) \tag{1}$$

and for new k, assuming that $F_0 = N(\mu = 0, \sigma_0^2 I)$,

$$\begin{aligned} \int p(x_i|\nu)\pi(\nu)d\nu &= C_1 \int \exp(-\|x_i - \nu\|^2/2) \exp(-\frac{\|\nu\|^2}{2\sigma_0^2})d\nu \\ &= C_1 \exp\left(-\frac{\|x_i\|^2}{2(1+\sigma_0^2)}\right) \end{aligned}$$

Therefore

$$p(z_i = K_{new} | x, z_{-i}, \mu) = C \frac{\alpha}{n - 1 + \alpha} g(x_i | 0, (1 + \sigma_0^2) I) \tag{2}$$

where $g(.|\mu, \sigma^2)$ is the pdf of $N(\mu, \sigma^2)$.

2.2.3 Prediction

For samples $t = 1, \cdots, T$, draw $\mu_*^{(t)} | \mu^{(t)}$ from CRP defined by Equation (1) in the question. Then predict $p(x_* | x_{1:n}) = \frac{1}{T} \sum_{t=1}^{T} N(\mu_*^{(t)}, I)$.