"Deriving Posterior Predictive Distribution of Gaussian Process Regression with RBF Kernel: Implementation and Analysis"

~Kevin Shah

Part 1:

To construct a surrogate function f(x) using Gaussian Process regression, we need to define the kernel function, also known as the covariance function. The kernel function's choice depends on the data's characteristics and the surrogate function's desired behaviour.

A commonly used kernel function for Gaussian Process regression is the squared exponential kernel, also known as the **Gaussian kernel or radial basis function (RBF) kernel**. It is defined as

$$k(x, x') = \sigma^2 * \exp(-(x - x')^2 / (2 * \ell^2))$$

where σ^2 is the variance parameter that controls the amplitude of the function, and ℓ is the length scale parameter that determines the smoothness and length of the correlation between data points.

- The positive-semidefinite property of the kernel function is crucial because it ensures that the resulting covariance matrix is positive-semidefinite. This property guarantees the existence of a valid Gaussian process and enables efficient computations using Cholesky decomposition.

- Cholesky decomposition is a numerical method used to factorize a positive-semidefinite matrix into the product of a lower triangular matrix and its conjugate transpose. In the context of Gaussian Process regression, Cholesky decomposition is applied to the covariance matrix of the training data to obtain the lower triangular matrix, which is then used to calculate the weights for the training points during prediction.

To **derive the posterior predictive distribution of the Gaussian Process**, we can follow these steps:

1. Define the Kernel Function (RBF):
   The RBF kernel is defined as:
   k(x, x') = exp(-||x - x'||^2 / (2 * length_scale^2))

2. Create the Kernel Matrix:
   Given the training data X with N data points, the kernel matrix K(X, X) is computed as follows:
   K(X, X) = [k(x_i, x_j)] for i, j = 1 to N

3. Add Noise to the Kernel Matrix (Optional):
   If there is noise in the data, create a noise matrix N of size N x N with diagonal elements equal to the noise variances.
   Update the kernel matrix as
   K(X, X) = K(X, X) + N

4. Perform Cholesky Decomposition:
   Perform the Cholesky decomposition on the kernel matrix K(X, X) to obtain the lower triangular matrix L:
   K(X, X) = L * L^T

5. Solve for Regression Weights:
   Solve the linear equations L^T * alpha = y, where L is the lower triangular matrix, alpha represents the regression weights, and y are the target values F(X).
   The solution can be obtained efficiently using forward and backward substitution.

6. Compute Mean and Covariance of Posterior Predictive Distribution:
   Given the test data X' with M data points, calculate the kernel matrix K(X', X) of size M x N:
   K(X', X) = [k(x'_i, x_j)] for i = 1 to M, j = 1 to N

7. Compute the mean vector:
   Given a test point x* and its corresponding kernel vector k(x*, X) of size 1 x N (N is the number of training points), and the regression weights vector alpha of size N x 1, the mean vector f* is computed as:
   f* = k(x*, X) * alpha

8. Compute the covariance matrix:
   Given a test point x* and its corresponding kernel matrix k(x*, x*) of size 1 x 1, the kernel vector k(x*, X) of size 1 x N, and the lower triangular matrix L obtained from the Cholesky decomposition, the covariance matrix K(X', X') of size M x M is computed as:
   K(X', X') = k(x*, x*) - k(x*, X) * (L^T)^(-1) * (L)^(-1) * k(x*, X)^T

These formulas give us the posterior predictive distribution of the Gaussian Process, where f* is the mean prediction at x* and v* is the associated variance.

f* = k(x*, X) * α
v* = k(x*, x*) - k(x*, X) * (L * L^T)^-1 * k(x*, X)^T

- Given the assumptions of a non-periodic, smooth, and differentiable function, the squared exponential kernel (RBF kernel) is a suitable choice as it captures these characteristics. However, depending on the specific problem and data, other kernel functions, such as the Matérn kernel, can also be used.

BONUS:

A.

When we talk *about how well a Gaussian Process extrapolates*, we are referring to its ability to make accurate predictions for input points that lie outside the range of the training data. In other words, it should be able to capture the underlying trends and patterns in the data and provide reasonable estimates for unseen data points.

B.

- In Gaussian Process regression, we don't typically perform *optimization* in the same way as in regular neural networks. The optimization step in neural networks involves adjusting the weights and biases through methods like gradient descent to minimize a predefined loss function. However, in Gaussian Process regression, there is no explicit optimization step for finding the best surrogate function.

- Instead, Gaussian Process regression provides a closed-form solution for the posterior predictive distribution, which takes into account the training data and provides a probabilistic estimation for new points. The process involves computing the covariance matrix, performing Cholesky decomposition, solving linear equations, and calculating the mean and variance of the predicted distribution. The properties of the kernel function and the training data directly determine the behaviour of the surrogate function.

- The performance of the Gaussian Process in extrapolation is largely influenced by the choice of the kernel function and its hyperparameters (e.g., length scale and variance). We can control the surrogate function's smoothness, flexibility, and generalization capabilities by selecting an appropriate kernel and tuning its parameters. However, there is no explicit optimization process involved in adjusting the surrogate function to improve extrapolation.

C.

- Regarding classification, Gaussian Processes can be adapted to perform probabilistic binary classification tasks using methods such as Gaussian Process Classification (GPC). GPC models estimate the posterior distribution over class labels rather than a continuous output. They leverage the principles of Gaussian Processes and apply them to classification problems.

- In Gaussian Process Classification, the training data consists of input points and corresponding binary class labels. Similar to Gaussian Process regression, a kernel function is chosen to model the relationships between the data points. However, the covariance matrix is modified to incorporate class labels and to capture the uncertainty in the predictions.

- During inference, Gaussian Process Classification provides the posterior probability of each class label for a given test point. The decision boundaries can be obtained by setting a threshold on the probabilities.

In summary, Gaussian Processes can be adjusted to perform classification tasks by using Gaussian Process Classification techniques. While Gaussian Process regression and classification involve different formulations and computations, they both leverage the principles of Gaussian Processes to provide probabilistic predictions for regression and classification problems, respectively.