Kernel Methods

Chapter 6

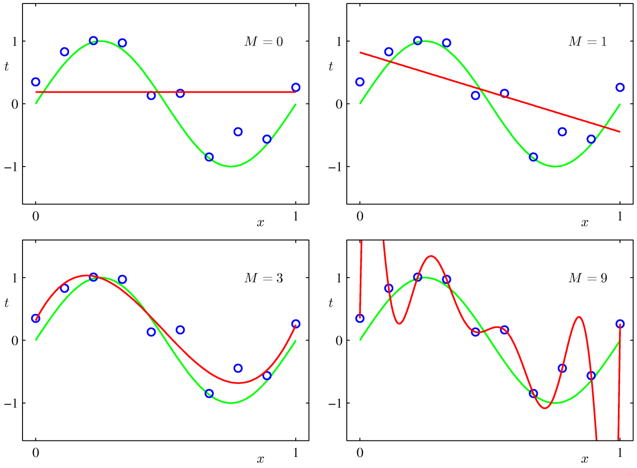
Prof. Reza Azadeh

University of Massachusetts Lowell

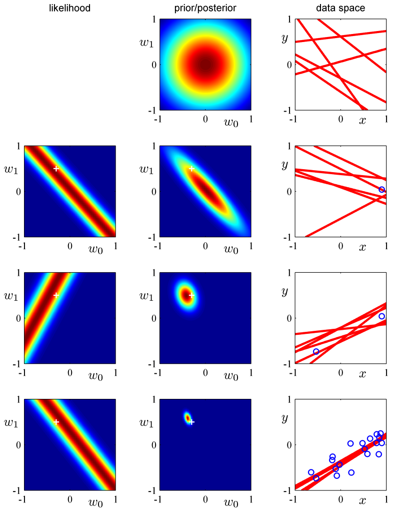
Previously

- In Chapters 3 and 4, we studied the linear models for regression and classification where we used a data set to learn a vector $\mathbf{w}$ of adaptive parameters.
- After learning phase, we construct $y(\mathbf{x}, \mathbf{w})$ and used it to predict new data points.
- The training data is then discarded, and predictions for new inputs are based purely on the learned parameter vector $\mathbf{w}$.
- This approach is also used in nonlinear parametric models such as neural networks.

Reminder: Linear models for regression



Reminder: Linear models for regression



Big picture

“ A Gaussian Process defines a prior over functions, which can be converted into a posterior over functions once we have seen some data.”

Non-parametric methods

- There is a class of techniques, in which the training data set, or a subset of it, is kept and used also during the prediction phase.
- These methods are referred to as *non-parametric* or *memory-based*, and involve storing the entire training set in order to make predictions of future data points.
- These methods require a metric to measure the similarity of any two vectors in input space known as a *kernel*.

Kernels

For models that use a fixed nonlinear feature space mapping $\boldsymbol{\phi}(\mathbf{x})$, the kernel function is given by

$$k(\mathbf{x}, \mathbf{x}') = \boldsymbol{\phi}(\mathbf{x})^\top \boldsymbol{\phi}(\mathbf{x}')$$

Kernel properties

- The kernel is a symmetric function of its arguments

$$k(\mathbf{x}, \mathbf{x}') = k(\mathbf{x}', \mathbf{x})$$

- We get the *linear kernel*, when we use the identity mapping for the feature space as $\boldsymbol{\phi}(\mathbf{x}) = \mathbf{x}$:

$$k(\mathbf{x}, \mathbf{x}') = \mathbf{x}^\top \mathbf{x}'$$

Kernel properties

- Kernels that are invariant to translations in input space are known as *stationary* kernels, for example

$$k(\mathbf{x}, \mathbf{x}') = k(\mathbf{x} - \mathbf{x}')$$

- Kernels that depend only on the magnitude of the distance between the arguments are called *homogeneous* kernels

$$k(\mathbf{x}, \mathbf{x}') = k(\|\mathbf{x} - \mathbf{x}'\|)$$

Constructing Kernels (1)

- One method for constructing valid kernels is to choose a feature space mapping $\boldsymbol{\phi}(\mathbf{x})$ and then use this to find the corresponding kernel.

Constructing Kernels (2)

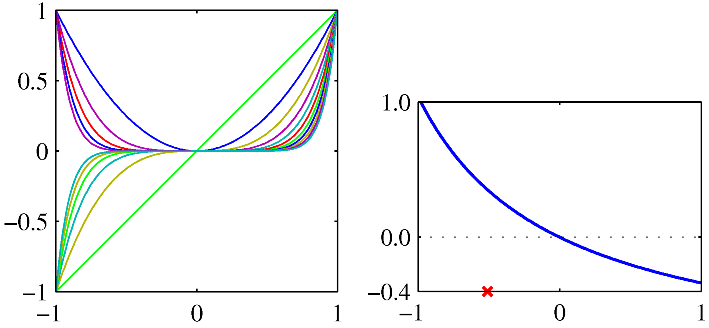


Figure: (left) a set of polynomial basis (right) $k(x, x') = \boldsymbol{\phi}(\mathbf{x})^\top \boldsymbol{\phi}(\mathbf{x}')$ is plotted as a function of x when x' is given by the red cross.

Constructing Kernels (3)

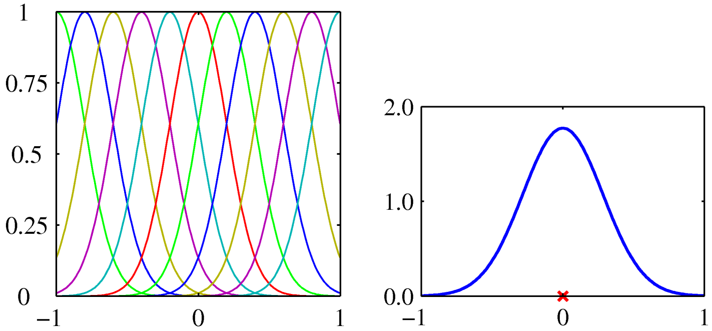


Figure: (left) a set of Gaussian basis (right) $k(x, x') = \boldsymbol{\phi}(\mathbf{x})^\top \boldsymbol{\phi}(\mathbf{x}')$ is plotted as a function of x when x' is given by the red cross.

Constructing Kernels (4)

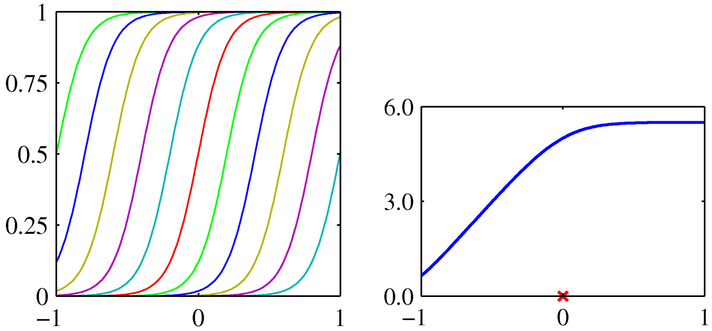
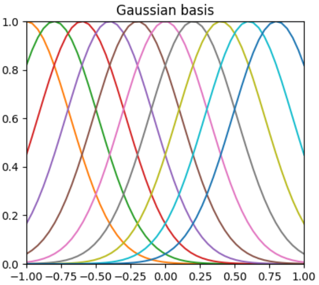


Figure: (left) a set of logistic sigmoids (right) $k(x, x') = \boldsymbol{\phi}(\mathbf{x})^\top \boldsymbol{\phi}(\mathbf{x}')$ is plotted as a function of x when x' is given by the red cross.

Constructing Kernels (5)



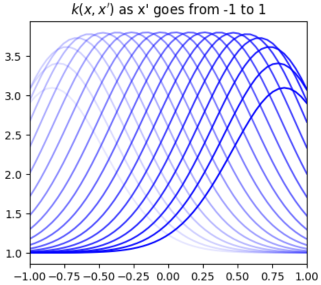


Figure: (left) a set of Gaussian basis (right) $k(x, x')$ is plotted as a function of x when x' moves from -1 to 1.

Constructing Kernels (6)

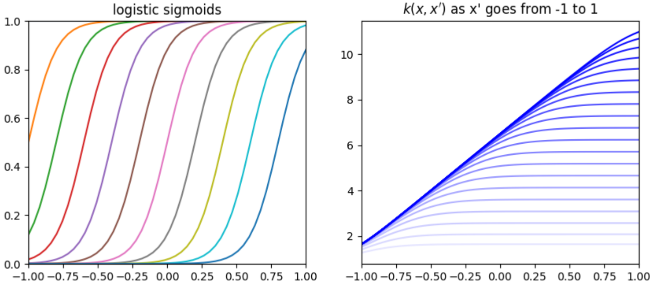


Figure: (left) a set of logistic sigmoids (right) $k(x, x')$ is plotted as a function of x when x' moves from -1 to 1.

Constructing Kernels (7)

- An alternative approach is to construct kernels directly. In this case, we must ensure that the constructed function is a valid kernel.
- This method requires the identification of $\phi(\mathbf{x})$ from the kernel which might not be very straight forward for complicated kernels.
- We need a simple way to test whether a function is a valid kernel without having to construct the basis function. explicitly.
- A necessary and sufficient condition for a function $k(\mathbf{x}, \mathbf{x}')$ to be a valid kernel is that the Gram matrix $\mathbf{K}$ whose elements are given by $k(\mathbf{x}_n, \mathbf{x}_m)$, should be positive semidefinite.

Reminder: Positive Definite Matrix (1)

- A matrix $\mathbf{A}$ is said to be **positive definite**, denoted by $\mathbf{A} \succ 0$, if

$$\mathbf{w}^\top \mathbf{A} \mathbf{w} > 0$$

  for all non-zero values of the vector $\mathbf{w}$.
- Equivalently, a positive definite matrix has $\lambda_i > 0$ for all of its eigenvalues.

Reminder: Positive Definite Matrix (2)

- Note that positive definite is not the same as all the elements being positive. For example, the following matrix is not positive definite:

$$\begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}$$

because it has eigenvalues $\lambda_1 \approx 5.37$ and $\lambda_2 \approx -0.37$.

Remidner: Positive Semidefinite Matrix

- A matrix $\mathbf{A}$ is said to be **positive semidefinite**, denoted by $\mathbf{A} \succeq 0$, if

$$\mathbf{w}^\top \mathbf{A} \mathbf{w} \geq 0$$

  for all non-zero values of the vector $\mathbf{w}$.
- Equivalently, a positive definite matrix has $\lambda_i \geq 0$ for all of its eigenvalues.

Constructing Kernels (8)

One powerful technique for constructing new kernels is to build them out of simpler kernels. This can be done using a set of proven properties.

Constructing Kernels (9)

Given valid kernels $k_1(\mathbf{x}, \mathbf{x}')$ and $k_2(\mathbf{x}, \mathbf{x}')$, the following new kernels are also valid.

$$
\begin{aligned}
k(\mathbf{x}, \mathbf{x}') &= ck_1(\mathbf{x}, \mathbf{x}') \\
k(\mathbf{x}, \mathbf{x}') &= f(\mathbf{x})k_1(\mathbf{x}, \mathbf{x}')f(\mathbf{x}') \\
k(\mathbf{x}, \mathbf{x}') &= q(k_1(\mathbf{x}, \mathbf{x}')) \\
k(\mathbf{x}, \mathbf{x}') &= \exp(k_1(\mathbf{x}, \mathbf{x}')) \\
k(\mathbf{x}, \mathbf{x}') &= k_1(\mathbf{x}, \mathbf{x}') + k_2(\mathbf{x}, \mathbf{x}') \\
k(\mathbf{x}, \mathbf{x}') &= k_1(\mathbf{x}, \mathbf{x}')k_2(\mathbf{x}, \mathbf{x}')
\end{aligned}
$$

where $c > 0$ is a constant, $f(.)$ is any function, and $q(.)$ is a polynomial with nonnegative coefficients. See more properties in the textbook.

Examples of constructing kernels

- Recall the linear kernel $k_1(\mathbf{x}, \mathbf{x}') = \mathbf{x}^\top \mathbf{x}'$.
- We can construct second order polynomial kernels by using the third property from the above list. We get:

$$k(\mathbf{x}, \mathbf{x}') = q(k_1(\mathbf{x}, \mathbf{x}')) = (\mathbf{x}^\top \mathbf{x}')^2$$

- This kernel only includes the quadratic terms. By adding a positive constant, we can make all linear and constant terms too

$$\begin{aligned} k_1(\mathbf{x}, \mathbf{x}') &= \mathbf{x}^\top \mathbf{x}' + c \\ k(\mathbf{x}, \mathbf{x}') &= q(k_1(\mathbf{x}, \mathbf{x}')) = (\mathbf{x}^\top \mathbf{x}' + c)^2 \end{aligned}$$

Examples of constructing kernels

- We can also construct higher order kernels

$$k(\mathbf{x}, \mathbf{x}') = q(k_1(\mathbf{x}, \mathbf{x}')) = (\mathbf{x}^\top \mathbf{x}')^M$$

- This can also be extended to include more lower order terms

$$\begin{aligned} k_1(\mathbf{x}, \mathbf{x}') &= \mathbf{x}^\top \mathbf{x}' + c \\ k(\mathbf{x}, \mathbf{x}') &= q(k_1(\mathbf{x}, \mathbf{x}')) = (\mathbf{x}^\top \mathbf{x}' + c)^M \end{aligned}$$

Linear regression revisited (1)

Consider a model defined in terms of a linear combination of M fixed basis functions given by the elements of the vector $\phi(\mathbf{x})$ so that

$$y(\mathbf{x}) = \mathbf{w}^\top \phi(\mathbf{x})$$

where $\mathbf{x}$ is the input vector and $\mathbf{w}$ is the M-dimensional weight vector. Now consider a prior distribution over $\mathbf{w}$ given by the isotropic Gaussian of the form

$$p(\mathbf{w}) = \mathcal{N}(\mathbf{w}|\mathbf{0}, \alpha^{-1}\mathbf{I})$$

where α represents the precision of the distribution.

Linear regression revisited (2)

- The probability distribution over $\mathbf{w}$ induces a probability distribution over function $y(\mathbf{x})$.
- We wish to evaluate this function at specific values of $\mathbf{x}$, for example at training data points $\mathbf{x}_1, \ldots, \mathbf{x}_N$.
- Therefore, we are interested in the joint distribution of the function values $y(\mathbf{x}_1), \ldots, y(\mathbf{x}_N)$, which we denote by vector $\bar{\mathbf{y}}$ with elements $y_n = y(\mathbf{x}_n)$ for $n = 1, \ldots, N$.
- We can write this vector as

$$\bar{\mathbf{y}} = \mathbf{\Phi}\mathbf{w}$$

where $\mathbf{\Phi}$ is the design matrix with elements $\Phi_{nk} = \phi_k(\mathbf{x}_n)$.

Linear regression revisited (3)

To find the probability distribution of $\bar{\mathbf{y}}$, we should note that $\bar{\mathbf{y}}$ is Gaussian since it is the linear combination of a set of Gaussian distributed variables.
We therefore need only to find its mean and covariance, which are

$$\mathbb{E}[\bar{\mathbf{y}}] = \boldsymbol{\Phi}\mathbb{E}[\mathbf{w}] = \mathbf{0}$$
$$\mathbb{V}[\bar{\mathbf{y}}] = \mathbb{E}[\bar{\mathbf{y}}\bar{\mathbf{y}}^\top] = \boldsymbol{\Phi}\mathbb{E}[\mathbf{w}\mathbf{w}^\top]\boldsymbol{\Phi}^\top = \frac{1}{\alpha}\boldsymbol{\Phi}\boldsymbol{\Phi}^\top = \mathbf{K}$$

where $\mathbf{K}$ is called the Gram matrix with elements

$$K_{nm} = k(x_n, x_m) = \frac{1}{\alpha}\phi(\mathbf{x}_n)^\top\phi(\mathbf{x}_m)$$

and $k(\mathbf{x}, \mathbf{x}')$ is the *kernel* function.

Linear regression revisited (4)

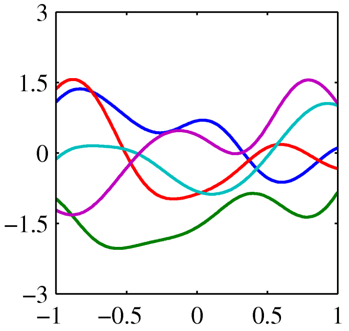
- This model provides us with a particular example of a Gaussian process.
- In general, a Gaussian process is defined as a probability distribution over function $y(\mathbf{x})$ such that the set of values of $y(\mathbf{x})$ evaluated at an arbitrary set of points $\mathbf{x}_1, \ldots, \mathbf{x}_N$ jointly have a Gaussian distribution. For a two dimensional input, this is also known as *Gaussian random field*.

Linear regression revisited (5)

- For this specific case, we choose a mean of zero, indicating we have no prior knowledge about the mean of the distribution.
- The covariance is evaluated at any two values of $\mathbf{x}$ given by the kernel function

$$\mathbb{E}[y(\mathbf{x}_n)y(\mathbf{x}_m)] = k(\mathbf{x}_n, \mathbf{x}_m)$$

Linear regression revisited (6)



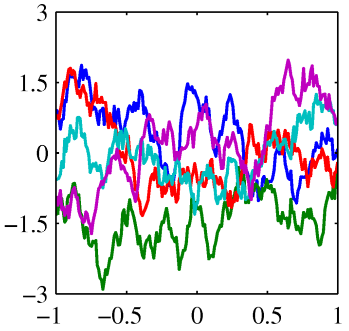


Figure: Samples from Gaussian process from a Gaussian kernel (left) and an exponential kernel (right)

Gaussian Processes for Regression (1)

We model the noise on the observed target values as

$$t_n = y_n + \epsilon_n$$

where $y_n = y(\mathbf{x}_n)$, and ϵ_n is a random noise variable whose value is chosen independently for each observation n.
Here we consider noise processes that have a Gaussian distribution, so that

$$p(t_n|y_n) = \mathcal{N}(t_n|y_n, \beta^{-1})$$

where β is a hyperparameter representing the precision of the noise.

Gaussian Processes for Regression (2)

Because the noise is independent for each data point, the joint distribution of the target values $\mathbf{t} = (t_1, \dots, t_N)^\top$ conditioned on the values of $\mathbf{y} = (y_1, \dots, y_N)^\top$ is given by an isotropic Gaussian of the form

$$p(\mathbf{t}|\mathbf{y}) = \mathcal{N}(\mathbf{t}|\mathbf{y}, \beta^{-1}\mathbf{I}_N)$$

where $\mathbf{I}_N$ denotes the $N \times N$ unit matrix.

Gaussian Processes for Regression (3)

From the definition of Gaussian process, the marginal distribution $p(\mathbf{y})$ is given by a Gaussian whose mean is zero and whose covariance is defined by a Gram matrix $\mathbf{K}$ so that

$$p(\mathbf{y}) = \mathcal{N}(\mathbf{y}|\mathbf{0}, \mathbf{K})$$

The kernel function that determines $\mathbf{K}$ is chosen to express the property that, for points $\mathbf{x}_n$ and $\mathbf{x}_m$ that are similar, the corresponding values $y(\mathbf{x}_n)$ and $y(\mathbf{x}_m)$ will be more strongly correlated than for dissimilar points.
Note that the notion of similarity is application dependent.

Gaussian Processes for Regression (3)

To find the marginal distribution $p(\mathbf{t})$, conditioned on the input values $\mathbf{x}_1, \ldots, \mathbf{x}_N$, we need to integrate over $\mathbf{y}$ (using results from Ch.2)

$$p(\mathbf{t}) = \int p(\mathbf{t}|\mathbf{y})p(\mathbf{y})d\mathbf{y} = \mathcal{N}(\mathbf{t}|\mathbf{0}, \mathbf{C})$$

where the covariance matrix $\mathbf{C}$ has elements

$$C(\mathbf{x}_n, \mathbf{x}_m) = k(\mathbf{x}_n, \mathbf{x}_m) + \beta^{-1}\delta_{nm}$$

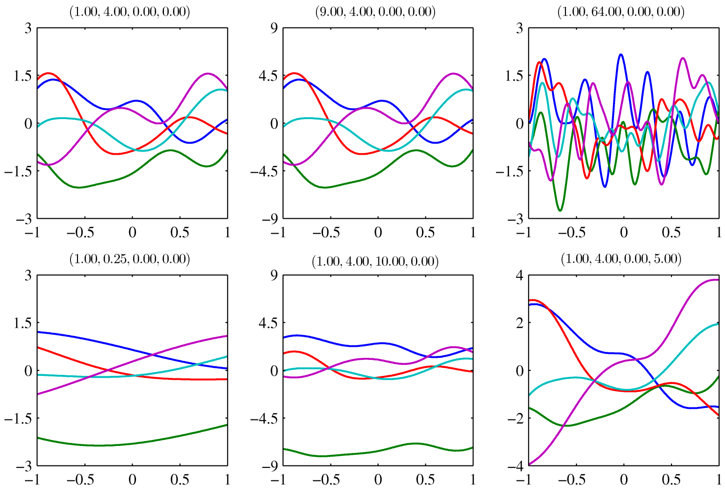
Note that this result indicates that the two sources of randomness, namely from $y(\mathbf{x})$ and ϵ are independent.

Gaussian Processes for Regression (4)

One wildly kernel function for Gaussian process regression is given by the exponential of the quadratic form, with the addition of linear and constant terms as

$$k(\mathbf{x}_n, \mathbf{x}_m) = \theta_0 \exp\{-\frac{\theta_1}{2}\|\mathbf{x}_n - \mathbf{x}_m\|^2\} + \theta_2 + \theta_3 \mathbf{x}_n^\top \mathbf{x}_m.$$

Gaussian Processes for Regression (4)



Gaussian Processes for Regression (5)

- So far, we have used the Gaussian process viewpoint to build a model of the joint distribution over sets of data points.
- Our goal in regression, however, is to make predictions of the target variables for new inputs, given a set of training data.
- Let us suppose that $\mathbf{t}_N = (t_1, \dots, t_N)^\top$, corresponding to input values $\mathbf{x}_1, \dots, \mathbf{x}_N$, comprise the observed training set, and our goal is to predict the target variable t_{N+1} for a new input vector $\mathbf{x}_{N+1}$.

Gaussian Processes for Regression (6)

- This requires that we evaluate the predictive distribution $p(t_{N+1}|\mathbf{t}_N)$. We have hidden the dependency on $\mathbf{x}_1, \dots, \mathbf{x}_{N+1}$.
- We start by writing down the joint distribution $p(\mathbf{t}_{N+1})$, where $\mathbf{t}_{N+1}$ denotes the vector $(t_1, \dots, t_N, t_{N+1})^\top$.

$$p(\mathbf{t}_{N+1}) = \mathcal{N}(\mathbf{t}_{N+1}|\mathbf{0}, \mathbf{C}_{N+1})$$

where $\mathbf{C}_{N+1}$ is an $(N+1) \times (N+1)$ covariance matrix with elements given by

$$C(\mathbf{x}_n, \mathbf{x}_m) = k(\mathbf{x}_n, \mathbf{x}_m) + \beta^{-1}\delta_{nm}$$

Gaussian Processes for Regression (7)

- Because this is Gaussian, we can partition the covariance matrix as follows:

$$\mathbf{C}_{N+1} = \begin{pmatrix} \mathbf{C}_N & \mathbf{k} \\ \mathbf{k}^\top & c \end{pmatrix}$$

  where $\mathbf{C}_N$ is the $N \times N$ covariance matrix with elements given by $C(\mathbf{x}_n, \mathbf{x}_m) = k(\mathbf{x}_n, \mathbf{x}_m) + \beta^{-1}\delta_{nm}$ for $n, m = 1, \ldots, N$, the vector $\mathbf{k}$ has elements $k(\mathbf{x}_n, \mathbf{x}_{N+1})$ for $n = 1, \ldots, N$, and the scalar $c = k(\mathbf{x}_{N+1}, \mathbf{x}_{N+1}) + \beta^{-1}$.

Gaussian Processes for Regression (8)

- Finally, it turns out that the conditional distribution $p(t_{N+1}|\mathbf{t})$ is a Gaussian distribution with mean and covariance given by

$$m(\mathbf{x}_{N+1}) = \mathbf{k}^\top \mathbf{C}_N^{-1} \mathbf{t}$$
$$\sigma^2(\mathbf{x}_{N+1}) = c - \mathbf{k}^\top \mathbf{C}_N^{-1} \mathbf{k}$$

- These are the key results that define Gaussian process regression.

Gaussian Processes for Regression (9)

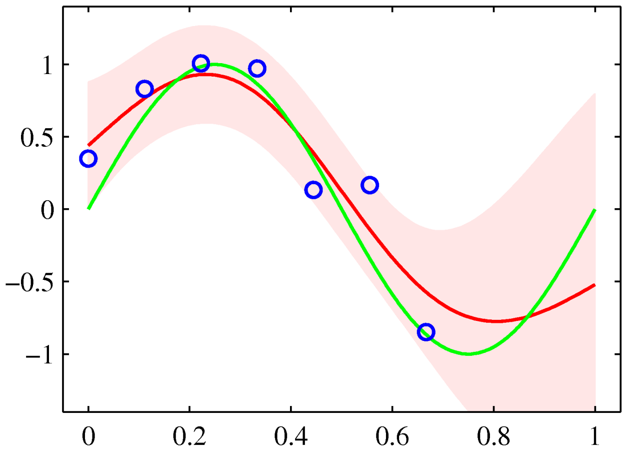


Figure: Gaussian process regression applied to the sinusoidal data set. (red) mean of the Gaussian process predictive distribution, (shaded area) $\pm 2\sigma$.

Learning the hyperparameters (1)

- The predictions of the GP model depends on the choice of covariance function.
- Instead of using a pre-defined function, we can learn the parameters of this function from data.
- To do this, we rely on the likelihood function $p(\mathbf{t}|\boldsymbol{\theta})$ where $\boldsymbol{\theta}$ denotes the set of hyperparameters.

Learning the hyperparameters (2)

The log likelihood for a Gaussian process regression model can be evaluated using the standard form of a multivariate Gaussian distribution

$$\ln p(\mathbf{t}|\boldsymbol{\theta}) = -\frac{1}{2}\ln|\mathbf{C}_N| - \frac{1}{2}\mathbf{t}^\top \mathbf{C}_N^{-1}\mathbf{t} - \frac{N}{2}\ln(2\pi)$$

we also need the gradient which is given by

$$\frac{\partial}{\partial\theta_i}\ln p(\mathbf{t}|\boldsymbol{\theta}) = -\frac{1}{2}\mathrm{Tr}(\mathbf{C}_N^{-1}\frac{\partial \mathbf{C}_N}{\partial\theta_i}) + \frac{1}{2}\mathbf{t}^\top \mathbf{C}_N^{-1}\frac{\partial \mathbf{C}_N}{\partial\theta_i}\mathbf{C}_N^{-1}\mathbf{t}$$

Learning the hyperparameters (3)

As mentioned before, $p(\mathbf{t}|\boldsymbol{\theta})$ is nonconvex and thus can have multiple maxima, therefore, it cannot be solved in a closed-form.
This optimization can be solved using gradient-based methods.

Gaussian Process for Classification (1)

- To extend the idea of Gaussian processes to classification, we must ensure that the posterior probabilities lie in the interval $(0, 1)$ (unlike regression).
- If we define a Gaussian process over a function $a(\mathbf{x})$ and then transform the function using a logistic sigmoid then we will obtain a non-Gaussian stochastic process over function $y(\mathbf{x})$.

Gaussian Process for Classification (2)

The Gaussian process prior for $\mathbf{a}_{N+1}$

$$p(\mathbf{a}_{N+1}) = \mathcal{N}(\mathbf{a}_{N+1}|\mathbf{0}, \mathbf{C}_{N+1})$$

where the vector $\mathbf{a}_{N+1} = (a(\mathbf{x}_1), \ldots, a(\mathbf{x}_{N+1}))^\top$, and the covariance matrix $\mathbf{C}_{N+1}$ has elements given by $C(\mathbf{x}_n, \mathbf{x}_m) = k(\mathbf{x}_n, \mathbf{x}_m) + \nu\delta_{nm}$. For a two class problem, we only need to find the prediction for one class

$$p(t_{N+1} = 1|\mathbf{t}_N) = \int p(t_{N+1} = 1|a_{N+1})p(a_{N+1}|\mathbf{t}_N)da_{N+1}$$

Gaussian Process for Classification (3)

The integral in analytically intractable, and we need to use approximation methods such as

- variational inference
- expectation propagation
- Laplace approximation

to calculate it.

Gaussian Process for Classification (4)

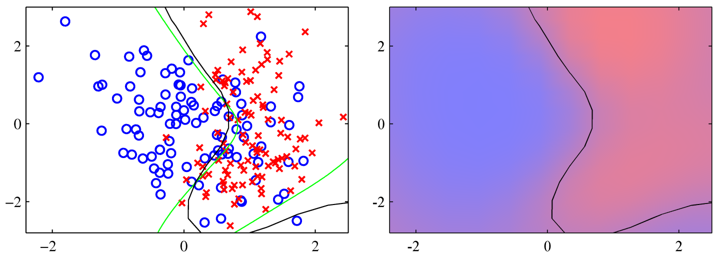


Figure: (left) the data and the decision boundary from the true distribution in green, and the decision boundary from the Gaussian process in black. (right) predicted posterior probability for the blue and the red classes.