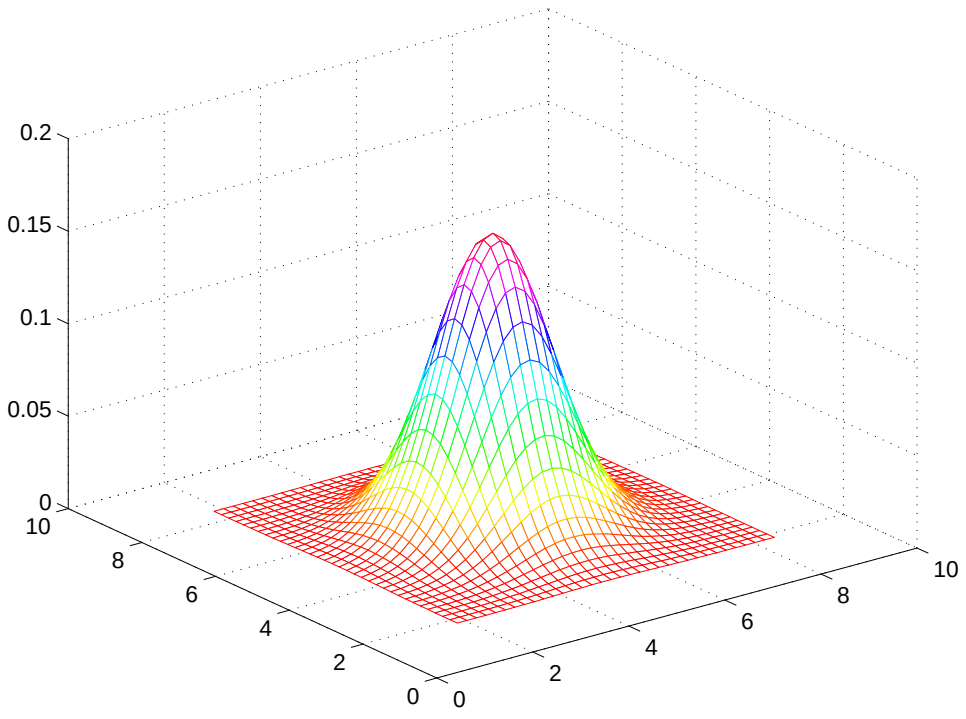
University of Cambridge
Engineering Part IIB

Paper 4F10: Statistical Pattern Processing

Handout 8: Gaussian Processes



Bill Byrne
bill.byrne@eng.cam.ac.uk
Michaelmas 2012

Outline

The next two lectures cover a range of topics motivating and describing Gaussian Processes. The following areas will be discussed:

- Regression problems
- Linear regression
  - Bayesian linear regression
- Regression/interpolation with basis functions
- Gaussian process regression
  - form of covariance function
- Classification with regression models
  - Bayesian classification
- Classification with Gaussian Processes

Regression

The majority of the techniques so far described have concentrated on classification rather than regression. Prediction/ interpolation are standard, commonly occurring, regression problems.

Here rather than mapping from an observation $\boldsymbol{x}$ to a class $y \in \{-1, 1\}$, we are interested in predicting a continuous value given a set of observations. Examples of this are:

- stock market prediction
- speech enhancement

The first part of these two lectures on Gaussian processes will look at regression. The application of Gaussian processes to classification will then be considered. For both cases supervised training will be assumed.

The regression (or prediction) is assumed to have two parts:

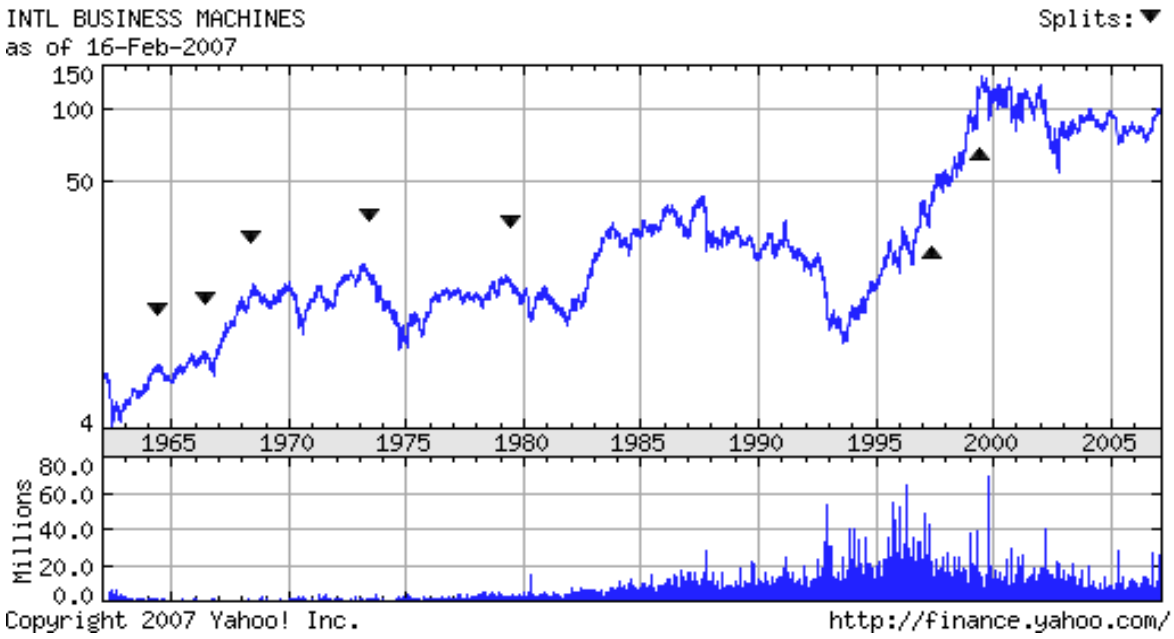
- systematic variation: can be accurately predicted if the underlying process is known ($f(\boldsymbol{x})$)
- random variation: the inherent "unpredictability" of the system.

These two parts will be modelled separately

$$y = f(\boldsymbol{x}) + \epsilon$$

where ϵ will be assumed to be Gaussian distributed, with a mean of zero and variance σ_{n}^2.

Stock Market Prediction



- Would like to make predictions and actions
- What value will IBM shares have today given the past performance?
  - IBM is now at $120.82 - $92.68 last year

    though not a linear rise!
- As well as the predicted value I am probably interested in the confidence of the prediction ...

Gaussian Process application

- Gaussian process approximations of stochastic differential equations.
- Prediction of deformed and annealed microstructures using Bayesian neural networks and Gaussian processes.
- Biomarker discovery in microarray gene expression data with Gaussian processes.
- Gasussian process model based predictive control.
- Gaussian processes for multiuser detection in cdma receivers.
- Learning a Gaussian process prior for automatically generating music playlists.
- GPPS: A Gaussian process positioning system for cellular networks.

For a wide-range of information and papers on Gaussian procesess see

`http://www.gaussianprocess.org/`

Basic Probability Revisited (again)

These lectures will make extensive use of the following standard probability concepts:

- continuous random variables: the RV can take any value within a, possibly, infinite, range. Associated probability density function satisfies

$$\int p(x)dx = 1; \quad p(x) \geq 0$$

- joint probability $p(x, y)$
- joint independence $p(x, y) = p(x)p(y)$
- conditional probability $p(x|y)$
- Bayes' rule

$$p(x|y) = \frac{p(x)p(y|x)}{p(y)}$$

- marginal probability

$$p(x) = \int_y p(x, y)dy$$

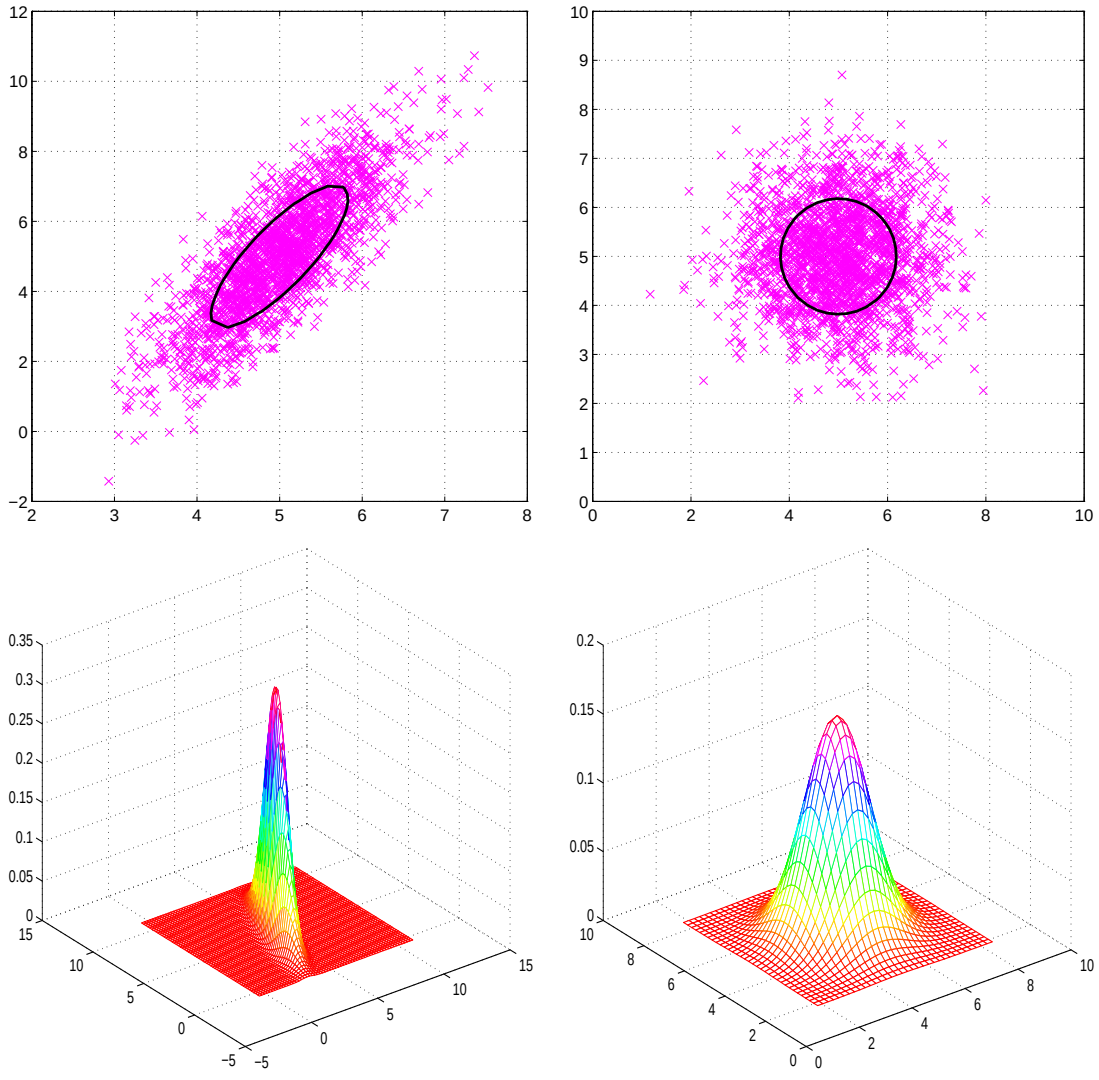
where the integral is over all possible values of y.

Multivariate Gaussian Distribution

For a d-dimensional data feature vector, $\boldsymbol{x}$, the multivariate Gaussian distribution is

$$p(\boldsymbol{x}) = \frac{1}{(2\pi)^{d/2}\left|\boldsymbol{\Sigma}\right|^{1/2}}\exp\left(-\frac{1}{2}\left(\boldsymbol{x}-\boldsymbol{\mu}\right)'\boldsymbol{\Sigma}^{-1}\left(\boldsymbol{x}-\boldsymbol{\mu}\right)\right)$$

$$\boldsymbol{\Sigma} = \begin{pmatrix} 3 & 1 \\ 1 & 0.5 \end{pmatrix} \qquad \boldsymbol{\Sigma} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$



Gaussian Distributions

Conditional Gaussian distribution:
Consider the joint distribution between vectors $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$

$$\begin{bmatrix} \boldsymbol{x}_1 \\ \boldsymbol{x}_2 \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \boldsymbol{\mu}_1 \\ \boldsymbol{\mu}_2 \end{bmatrix}, \begin{bmatrix} \boldsymbol{\Sigma}_{11} & \boldsymbol{\Sigma}_{12} \\ \boldsymbol{\Sigma}_{21} & \boldsymbol{\Sigma}_{22} \end{bmatrix} \right)$$

We are interested in the conditional distribution, which itself is Gaussian

$$\boldsymbol{x}_1 | \boldsymbol{x}_2 \sim \mathcal{N}(\boldsymbol{\mu}_{1|2}, \boldsymbol{\Sigma}_{1|2})$$

where

$$\begin{aligned} \boldsymbol{\mu}_{1|2} &= \boldsymbol{\mu}_1 + \boldsymbol{\Sigma}_{12}\boldsymbol{\Sigma}_{22}^{-1}(\boldsymbol{x}_2 - \boldsymbol{\mu}_2) \\ \boldsymbol{\Sigma}_{1|2} &= \boldsymbol{\Sigma}_{11} - \boldsymbol{\Sigma}_{12}\boldsymbol{\Sigma}_{22}^{-1}\boldsymbol{\Sigma}_{21} \end{aligned}$$

Product of Gaussian distributions:
Consider the two distributions

$$p_1(\boldsymbol{x}) = \mathcal{N}(\boldsymbol{x}; \boldsymbol{\mu}_1, \boldsymbol{\Sigma}_1), \quad p_2(\boldsymbol{x}) = \mathcal{N}(\boldsymbol{x}; \boldsymbol{\mu}_2, \boldsymbol{\Sigma}_2)$$

The product is an un-normalised Gaussian

$$p_1(\boldsymbol{x})p_2(\boldsymbol{x}) \propto \mathcal{N}(\boldsymbol{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma})$$

where

$$\begin{aligned} \boldsymbol{\mu} &= \boldsymbol{\Sigma}(\boldsymbol{\Sigma}_1^{-1}\boldsymbol{\mu}_1 + \boldsymbol{\Sigma}_2^{-1}\boldsymbol{\mu}_2) \\ \boldsymbol{\Sigma} &= \left(\boldsymbol{\Sigma}_1^{-1} + \boldsymbol{\Sigma}_2^{-1}\right)^{-1} \end{aligned}$$

Linear Interpolation

Consider the supervised training data of n samples, each with an observation $\boldsymbol{x}_i$ and output y_i. The regression function $f(\boldsymbol{x})$ is linear if defined as

$$f(\boldsymbol{x}) = \boldsymbol{w}'\boldsymbol{x}$$

and the target value has Gaussian noise so that

$$y = f(\boldsymbol{x}) + \epsilon$$

where $\epsilon \sim \mathcal{N}(0, \sigma_{\mathtt{n}}^2)$.

For a given value of $\boldsymbol{w}$ the likelihood of the outputs can be expressed as

$$p(y_1, \ldots, y_n | \boldsymbol{x}_1, \ldots, \boldsymbol{x}_n, \boldsymbol{w}) = \prod_{i=1}^{n} p(y_i | \boldsymbol{x}_i, \boldsymbol{w}) = \mathcal{N}(\boldsymbol{y}; \boldsymbol{X}'\boldsymbol{w}, \sigma_{\mathtt{n}}^2 \mathbf{I})$$

where $\mathbf{I}$ is the $n \times n$ identity matrix and

$$\boldsymbol{X} = \begin{bmatrix} \boldsymbol{x}_1 & \ldots & \boldsymbol{x}_n \end{bmatrix}, \quad \boldsymbol{y} = \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix}$$

The value of $\boldsymbol{w}$ can then be found by maximising this likelihood. This is equivalent to minimising the least squares cost function

$$\min_{\boldsymbol{w}} \left\{ \sum_{i=1}^{n} (y_i - \boldsymbol{w}'\boldsymbol{x}_i)^2 \right\} = \min_{\boldsymbol{w}} \left\{ ||\boldsymbol{y} - \boldsymbol{X}'\boldsymbol{w}||^2 \right\}$$

which can be solved using the pseudo-inverse

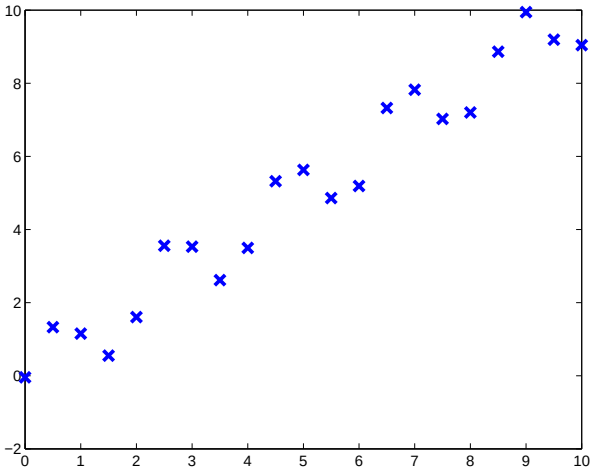
$$\hat{\boldsymbol{w}} = (\boldsymbol{X}\boldsymbol{X}')^{-1}\boldsymbol{X}\boldsymbol{y}$$

Example of Linear Regression

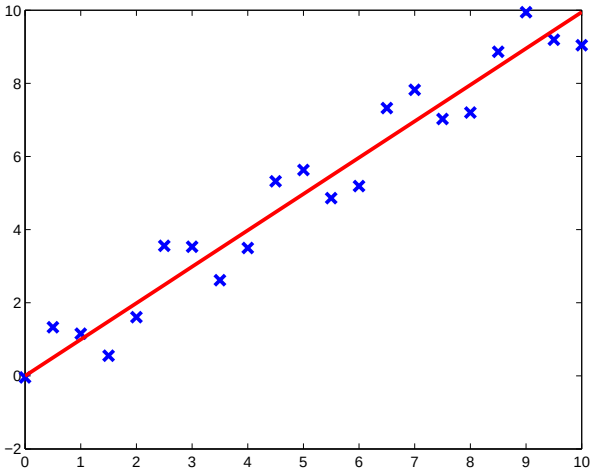
Generate data using

$$y = x + \sin(3x) + \epsilon$$

where $\epsilon \sim \mathcal{N}(0, 0.01)$.



A set of points generated using this function



Linear regression (interpolation/prediction) using this data

$$f(x) = 0.9948x$$

Bayesian Parameter Estimation

The parameter estimation schemes have assumed that there is a single estimate of the model parameters that can be obtained from the given training data. As this is only one possible set of training data (from all possible sets), then error bars are sometimes used to reflect this uncertainty.

In contrast a Bayesian scheme for estimating the model parameters, estimates a distribution over the model parameters. This distribution represents the uncertainty.

These two approaches are often referred to as

- Frequentist: linear regression prediction from $\boldsymbol{x}$ is

$$\hat{y} = \boldsymbol{x}'\hat{\boldsymbol{w}}$$

  where $\hat{\boldsymbol{w}}$ is the ML estimate (possibly include error bars)

- Bayesian: the distribution over the prediction is

$$p(y|\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{X}) = \int p(y|\boldsymbol{x}, \boldsymbol{w}) p(\boldsymbol{w}|\boldsymbol{y}, \boldsymbol{X}) d\boldsymbol{w}$$

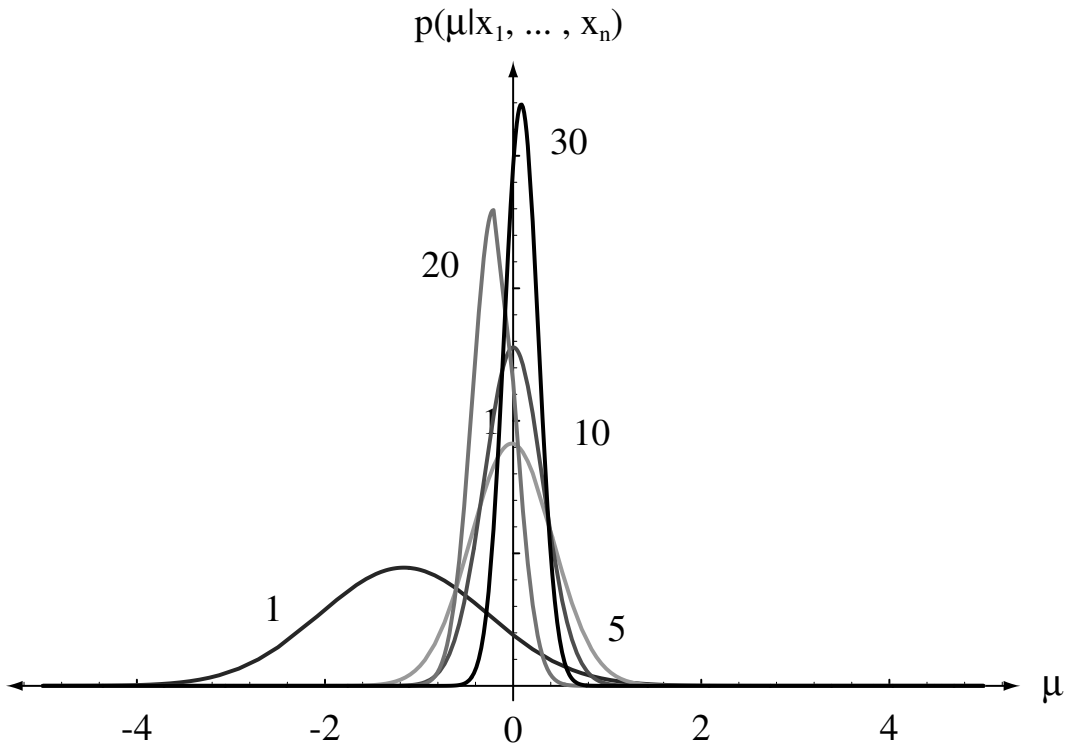
  where $p(\boldsymbol{w}|\boldsymbol{y}, \boldsymbol{X})$ is the distribution over the model parameters given the training data. The point estimate can be selected using for example the mean of the output distribution.

Bayesian approaches are becoming increasingly popular as the additional computational load of the marginalisation (integration) is becoming less of an issue through the use of approximations (and more computers!).

Gaussian Mean "Estimation"

As an aside Bayesian estimation can be applied to model parameters. Consider the Bayesian estimation of the mean.

The figure below (from DHS) shows how the posterior distribution for estimating the mean of a Gaussian distribution as the number of data samples increases.



- The posterior becomes more sharply peaked (reduced variance) and the MAP estimate (the mode of the distribution) gradually changes.
- As the number of samples increases the MAP estimate will tend towards the ML estimate.

Bayesian Linear Regression

A Bayesian version of linear regression may be used. Here a distribution over the weight vector is estimated and then marginalised over to yield the prediction. "Standard" linear regression can be written as

$$p(y|\boldsymbol{x}, \boldsymbol{X}, \boldsymbol{y}) = \int p(y|\boldsymbol{x}, \boldsymbol{w}) p(\boldsymbol{w}|\boldsymbol{X}, \boldsymbol{y}) d\boldsymbol{w}$$

where

$$\begin{aligned} p(y|\boldsymbol{x}, \boldsymbol{w}) &= \mathcal{N}(y; \boldsymbol{w}'\boldsymbol{x}, \sigma_{\mathrm{n}}^2) \\ p(\boldsymbol{w}|\boldsymbol{X}, \boldsymbol{y}) &= \delta(\boldsymbol{w} - \hat{\boldsymbol{w}}) \end{aligned}$$

This can then be written as

$$p(y|\boldsymbol{x}, \boldsymbol{X}, \boldsymbol{y}) = \mathcal{N}(y; \hat{\boldsymbol{w}}'\boldsymbol{x}, \sigma_{\mathrm{n}}^2)$$

If only the deterministic part of the regression progress is considered

$$f(\boldsymbol{x}) = \hat{\boldsymbol{w}}'\boldsymbol{x}$$

More generally the full distribution can be used. From the training data, $\boldsymbol{y}$ and $\boldsymbol{X}$, the distribution of the weights is

$$p(\boldsymbol{w}|\boldsymbol{y}, \boldsymbol{X}) = \frac{p(\boldsymbol{y}|\boldsymbol{X}, \boldsymbol{w}) p(\boldsymbol{w})}{p(\boldsymbol{y}|\boldsymbol{X})}$$

the denominator is independent of the weight vector so

$$p(\boldsymbol{w}|\boldsymbol{y}, \boldsymbol{X}) \propto p(\boldsymbol{y}|\boldsymbol{X}, \boldsymbol{w}) p(\boldsymbol{w})$$

Weight Vector Posterior Distribution

If the weights vector has a Gaussian prior then

$$\begin{aligned} p(\boldsymbol{y}|\boldsymbol{X},\boldsymbol{w}) &= \mathcal{N}(\boldsymbol{y};\boldsymbol{X}'\boldsymbol{w},\sigma_{\mathtt{n}}^2\mathbf{I}) \\ \boldsymbol{w} &\sim \mathcal{N}(\mathbf{0},\sigma_{\mathtt{w}}^2\mathbf{I}) \end{aligned}$$

The distribution of the weights can then be written as

$$\log(p(\boldsymbol{w}|\boldsymbol{y},\boldsymbol{X})) \propto -\frac{1}{2\sigma_{\mathtt{n}}^2}(\boldsymbol{y}-\boldsymbol{X}'\boldsymbol{w})'(\boldsymbol{y}-\boldsymbol{X}'\boldsymbol{w})) - \frac{1}{2\sigma_{\mathtt{w}}^2}\boldsymbol{w}'\boldsymbol{w}$$

As both are Gaussian distributed

$$p(\boldsymbol{w}|\boldsymbol{y},\boldsymbol{X}) = \mathcal{N}(\boldsymbol{w};\boldsymbol{\mu}_w,\boldsymbol{\Sigma}_w)$$

where

$$\begin{aligned} \boldsymbol{\mu}_w &= \frac{1}{\sigma_{\mathtt{n}}^2}\left(\frac{1}{\sigma_{\mathtt{n}}^2}\boldsymbol{X}\boldsymbol{X}'+\frac{1}{\sigma_{\mathtt{w}}^2}\mathbf{I}\right)^{-1}\boldsymbol{X}\boldsymbol{y} \\ \boldsymbol{\Sigma}_w &= \left(\frac{1}{\sigma_{\mathtt{n}}^2}\boldsymbol{X}\boldsymbol{X}'+\frac{1}{\sigma_{\mathtt{w}}^2}\mathbf{I}\right)^{-1} \end{aligned}$$

Note: as $\sigma_{\mathtt{w}}^2 \to \infty$, then $\boldsymbol{\mu}_w \to \hat{\boldsymbol{w}}$ the ML estimate.

This has so far only considered the training.

What about regression/prediction?

Bayesian Linear Regression (cont)

The joint distribution of all the weights is Gaussian. Regression can be based on this weight posterior.

Given an observation $\boldsymbol{x}$, and the training data $\boldsymbol{X}$ and $\boldsymbol{y}$, what is the distribution of the output value y.

- First examine the deterministic part: $f(\boldsymbol{x})$

$$\begin{aligned} p(f(\boldsymbol{x})|\boldsymbol{x}, \boldsymbol{X}, \boldsymbol{y}) &= \int p(f(\boldsymbol{x})|\boldsymbol{x}, \boldsymbol{w}) p(\boldsymbol{w}|\boldsymbol{X}, \boldsymbol{y}) d\boldsymbol{w} \\ &= \mathcal{N}(f(\boldsymbol{x}); \boldsymbol{x}'\boldsymbol{\mu}_w, \boldsymbol{x}'\boldsymbol{\Sigma}_w\boldsymbol{x}) \end{aligned}$$

  since $f(\boldsymbol{x})$ is a linear transform of the weight vector $\boldsymbol{w}$. This distribution over the function $f(\boldsymbol{x})$ includes all the information that can be extracted from the training data and the observation, $\boldsymbol{x}$.

- The distribution of the target value y is easy to find - this is the deterministic part, $f(\boldsymbol{x})$, plus the noise ϵ, as these are independent and Gaussian

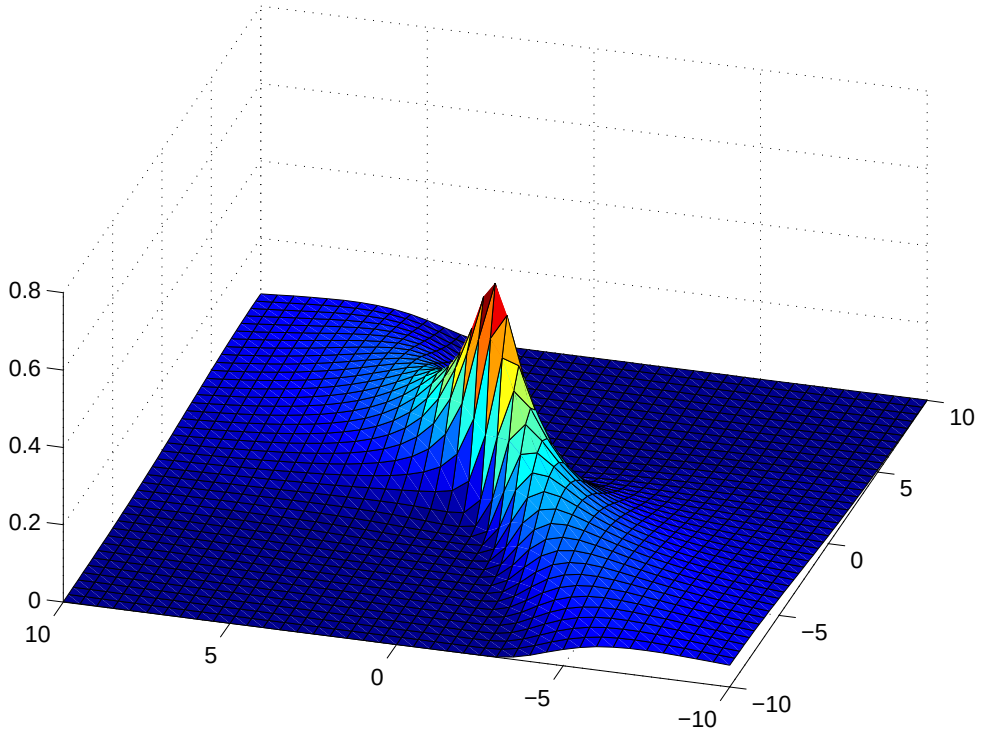
$$\begin{aligned} p(y|\boldsymbol{x}, \boldsymbol{X}, \boldsymbol{y}) &= \int p(y|f(\boldsymbol{x}) p(f(\boldsymbol{x})|\boldsymbol{X}, \boldsymbol{y}, \boldsymbol{x}) df(\boldsymbol{x}) \\ &= \mathcal{N}(y; \boldsymbol{x}'\boldsymbol{\mu}_w, \boldsymbol{x}'\boldsymbol{\Sigma}_w\boldsymbol{x} + \sigma_{\mathrm{n}}^2) \end{aligned}$$

Thus the posterior distribution of the weights are Gaussian as is the prediction of the observations. This is not surprising as the relationship the output and the function is linear. If this relationship is non-linear then the posterior and prediction will no longer be Gaussian.

Bayesian Linear Regression Example

Consider a simple example with

- 1-dimensional observations, x, $\sigma_{\mathrm{n}}^2 = 0.5$
- From training data $\mu_w = 1$, $\sigma_w^2 = 0.5$



$$p(y|x, \boldsymbol{X}, \boldsymbol{y}) = \mathcal{N}(y; x\mu_w, x^2\sigma_w^2 + \sigma_{\mathrm{n}}^2)$$

Above plot shows distribution over the input/output-space

- Regression forced to go through zero
  - variance at this point σ_{n}^2
- Variance increases as we move away from the origin
- The increase in variance depends on σ_w^2
  - decreases as $\sum_{i=1}^n x_i^2$ increases

Marginal Likelihood

In addition to using the training data for prediction it will also be useful to look at the marginal likelihood (compare this to the evidence)

$$p(\boldsymbol{y}|\boldsymbol{X}) = \int p(\boldsymbol{y}|\boldsymbol{X}, \boldsymbol{w})p(\boldsymbol{w})d\boldsymbol{w}$$

This is useful as the values of $\boldsymbol{w}$ are not observed, whereas $\boldsymbol{X}$ and $\boldsymbol{y}$ are. Effectively $\boldsymbol{w}$ is treated as a latent variable.

Using the distributions defined on the previous slide, it is clear that the marginal likelihood is also Gaussian distributed

$$\begin{aligned} p(\boldsymbol{y}|\boldsymbol{X}) &= \int \mathcal{N}(\boldsymbol{y}; \boldsymbol{X}'\boldsymbol{w}, \sigma_{\mathtt{n}}^2\mathbf{I})\mathcal{N}(\boldsymbol{w}; \mathbf{0}, \sigma_{\mathtt{w}}^2\mathbf{I})d\boldsymbol{w} \\ &= \mathcal{N}(\boldsymbol{y}; \mathbf{0}, \sigma_{\mathtt{w}}^2\boldsymbol{X}'\boldsymbol{X} + \sigma_{\mathtt{n}}^2\mathbf{I}) \end{aligned}$$

Note this has made use of the following matrix equality

$$\frac{1}{\sigma_{\mathtt{n}}^2}\mathbf{I} - \frac{1}{\sigma_{\mathtt{n}}^2}\boldsymbol{X}'\left(\frac{1}{\sigma_{\mathtt{n}}^2}\boldsymbol{X}\boldsymbol{X}' + \frac{1}{\sigma_{\mathtt{w}}^2}\mathbf{I}\right)^{-1}\boldsymbol{X}\frac{1}{\sigma_{\mathtt{n}}^2} = \left(\sigma_{\mathtt{w}}^2\boldsymbol{X}'\boldsymbol{X} + \sigma_{\mathtt{n}}^2\mathbf{I}\right)^{-1}$$

The marginal likelihood can be used as a mechanism for finding the parameters $\sigma_{\mathtt{w}}^2$ and $\sigma_{\mathtt{n}}^2$.

Interpolation using Basis Functions

Linear interpolation is limited as the relationship between the observations and the output (or target value) is constrained to be linear. An alternative is to perform interpolation using a set of basis functions.

Consider a basis function of the form $\phi(||\boldsymbol{x} - \boldsymbol{x}_i||)$, where $\phi()$ is some non-linear function and $||\boldsymbol{x} - \boldsymbol{x}_i||$ is a distance of the vector $\boldsymbol{x}$ from the prototype vector $\boldsymbol{x}_i$.

For the case of n training examples each being used as a prototype, the mapping can be defined as

$$f(\boldsymbol{x}) = \sum_{i=1}^{n} w_i \phi(||\boldsymbol{x} - \boldsymbol{x}_i||) = \boldsymbol{\phi}(\boldsymbol{x})' \boldsymbol{w}$$

where

$$\boldsymbol{\phi}(\boldsymbol{x}) = \begin{bmatrix} \phi(||\boldsymbol{x} - \boldsymbol{x}_1||) & \dots & \phi(||\boldsymbol{x} - \boldsymbol{x}_n||) \end{bmatrix}'$$

The output value is again considered to be

$$y = f(\boldsymbol{x}) + \epsilon$$

The values for $\boldsymbol{w}$ needs to estimated. Following the standard linear interpolation example, for the training data

$$\boldsymbol{\Phi} = \begin{bmatrix} \phi(||\boldsymbol{x}_1 - \boldsymbol{x}_1||) & \dots & \phi(||\boldsymbol{x}_1 - \boldsymbol{x}_n||) \\ \vdots & \ddots & \vdots \\ \phi(||\boldsymbol{x}_n - \boldsymbol{x}_1||) & \dots & \phi(||\boldsymbol{x}_n - \boldsymbol{x}_n||) \end{bmatrix} = \begin{bmatrix} \boldsymbol{\phi}(\boldsymbol{x}_1)' \\ \vdots \\ \boldsymbol{\phi}(\boldsymbol{x}_n)' \end{bmatrix}$$

So for the training data

$$\boldsymbol{y} = \boldsymbol{\Phi} \boldsymbol{w} + \boldsymbol{\epsilon}$$

Interpolation using Basis Functions (2)

If the inverse $\boldsymbol{\Phi}^{-1}$ exists then the ML estimate is

$$\hat{\boldsymbol{w}} = \boldsymbol{\Phi}^{-1}\boldsymbol{y}$$

It has been shown that for a large class of functions $\phi()$ if the set of points $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n$ is distinct then $\boldsymbol{\Phi}^{-1}$ exists.

With the weights defined above the function $f(\boldsymbol{x})$ represents a continuous function that passes through all the target points, y_i - this is exact interpolation.

There are number of standard forms of $\phi()$ used:

Non-linearity	Form ($z = (\|\|\boldsymbol{x} - \boldsymbol{x}_p\|\|)/\sigma$)
Gaussian	$\exp(-z^2/2)$
Exponential	$\exp(-z)$
Quadratic	$z^2 + \alpha z + \beta$
Inverse Quadratic	$\frac{1}{1+z^2}$
Thin plate spine	$z^\alpha \log(z)$
Trigonometric	$\sin(z)$

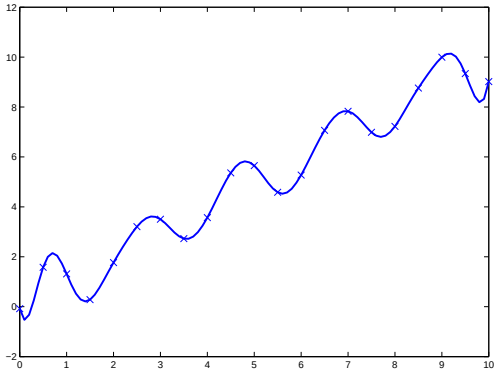
σ is a variable that controls the smoothness of the interpolation function.

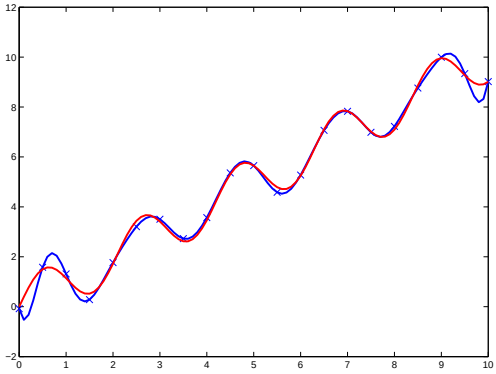
Some of the functions are localised basis functions. These have the property that $\phi(z) \rightarrow 0$ as $z \rightarrow \infty$.

Basis Function Example

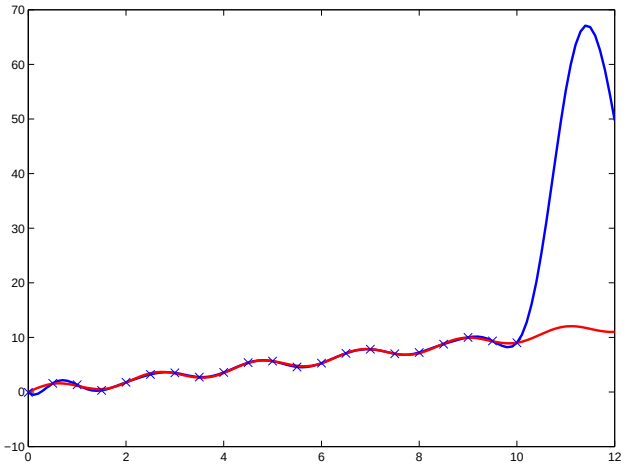
Interpolation was performed using a Gaussian basis function

$$\phi(||x - x_i||) = \exp\left(-\frac{1}{2}(x - x_i)^2\right)$$





Using the Gaussian basis finds the non-linearities and models the function well (exactly models the training points!).



However prediction is not too good!

Useful to use a Bayesian approach to directly get distribution

Bayesian Regression with Basis Functions

The previous analysis of Bayesian Linear regression can be applied to regression with basis functions.

Again, let the prior distribution for the weight vector be Gaussian distributed

$$p(\boldsymbol{w}) = \mathcal{N}(\boldsymbol{w}; \mathbf{0}, \sigma_{\mathtt{w}}^2 \mathbf{I})$$

We need the posterior weight distribution. Now

$$p(\boldsymbol{y}|\boldsymbol{X}, \boldsymbol{w}) = \mathcal{N}(\boldsymbol{y}; \boldsymbol{\Phi}\boldsymbol{w}, \sigma_{\mathtt{n}}^2 \mathbf{I})$$

From the linear case (equating $\boldsymbol{\Phi}$ and $\boldsymbol{X}'$)

$$p(\boldsymbol{w}|\boldsymbol{X}, \boldsymbol{y}) = \mathcal{N}(\boldsymbol{y}; \boldsymbol{\mu}_w, \boldsymbol{\Sigma}_w)$$

where

$$\begin{aligned} \boldsymbol{\mu}_w &= \frac{1}{\sigma_{\mathtt{n}}^2}\left(\frac{1}{\sigma_{\mathtt{n}}^2}\boldsymbol{\Phi}'\boldsymbol{\Phi} + \frac{1}{\sigma_{\mathtt{w}}^2}\mathbf{I}\right)^{-1}\boldsymbol{\Phi}'\boldsymbol{y} \\ \boldsymbol{\Sigma}_w &= \left(\frac{1}{\sigma_{\mathtt{n}}^2}\boldsymbol{\Phi}'\boldsymbol{\Phi} + \frac{1}{\sigma_{\mathtt{w}}^2}\mathbf{I}\right)^{-1} \end{aligned}$$

Thus a Bayesian form of non-linear regression may be addressed in the same fashion as linear regression, by using basis functions.

Training Output Covariance Matrix

It is interesting (and related to Gaussian Processes) to look at the covariance matrix of the training outputs, $\boldsymbol{y}$. The covariance matrix of $\boldsymbol{y}$, $\boldsymbol{\Sigma}_y$, can then be written as

$$\boldsymbol{\Sigma}_y = \mathcal{E}\left\{\boldsymbol{y}\boldsymbol{y}'\right\} - \mathcal{E}\left\{\boldsymbol{y}\right\}\mathcal{E}\left\{\boldsymbol{y}\right\}'$$

By inspection the mean is zero ($\boldsymbol{w}$ has zero mean prior, as is ϵ)

$$\begin{aligned}\boldsymbol{\Sigma}_y &= \mathcal{E}\left\{\boldsymbol{y}\boldsymbol{y}'\right\} \\ &= \mathcal{E}\left\{(\boldsymbol{\Phi}\boldsymbol{w}+\epsilon)(\boldsymbol{\Phi}\boldsymbol{w}+\epsilon)'\right\} \\ &= \sigma_{\mathtt{w}}^2\boldsymbol{\Phi}\boldsymbol{\Phi}' + \sigma_{\mathtt{n}}^2\mathbf{I}\end{aligned}$$

(same form as obtained for the marginal likelihood)

The target values are all jointly Gaussian under these assumptions. The same process can be applied to the weights posterior distribution. These are both examples of Gaussian Processes.

Interestingly element i,j element of the covariance matrix, $\boldsymbol{\Sigma}_y$, has the form

$$\sigma_{\mathtt{w}}^2\boldsymbol{\phi}(\boldsymbol{x}_i)'\boldsymbol{\phi}(\boldsymbol{x}_j) + \sigma_{\mathtt{n}}^2\delta_{ij}$$

where

$$\delta_{ij} = \begin{cases} 1, & i = j \\ 0, & \text{otherwise} \end{cases}$$

Gaussian Processes

The definition is:

> A Gaussian Process is a collection of random variables, any finite number of which have Gaussian distributions.

A Gaussian Process is completely specified by:

- mean function:

$$m(\boldsymbol{x}) = \mathcal{E}\{f(\boldsymbol{x})\}$$

- covariance function:

$$k(\boldsymbol{x}, \boldsymbol{x}_j) = \mathcal{E}\{(f(\boldsymbol{x}) - m(\boldsymbol{x}))(f(\boldsymbol{x}_j) - m(\boldsymbol{x}_j))\}$$

A Gaussian Process for two elements can be written as

$$\begin{bmatrix} f(\boldsymbol{x}_i) \\ f(\boldsymbol{x}_j) \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} m(\boldsymbol{x}_i) \\ m(\boldsymbol{x}_j) \end{bmatrix}, \begin{bmatrix} k(\boldsymbol{x}_i, \boldsymbol{x}_i) & k(\boldsymbol{x}_i, \boldsymbol{x}_j) \\ k(\boldsymbol{x}_i, \boldsymbol{x}_j) & k(\boldsymbol{x}_j, \boldsymbol{x}_j) \end{bmatrix}\right)$$

For simplicity the mean function will be taken as zero. Note for linear regression $f(\boldsymbol{x}) = \boldsymbol{w}'\boldsymbol{x}$.

Initially GPs will be considered for regression. This will have a form similar to Bayesian regression. The distribution of the output (including the random variation)

$$p(y|\boldsymbol{X}, \boldsymbol{x}, \boldsymbol{y}) = \int p(y|f(\boldsymbol{x}))p(f(\boldsymbol{x})|\boldsymbol{X}, \boldsymbol{x}, \boldsymbol{y})df(\boldsymbol{x})$$

where $f(\boldsymbol{x})$ is the systematic prediction.

Regression using Gaussian Processes

We would like to do prediction given that the outputs are all jointly Gaussian.

Initially let's examine obtaining the best prediction process $f(\boldsymbol{x})$ given the training data.

- Assume that the prediction process is jointly Gaussian distributed with the training outputs.
- As in basis function regression let the covariance function be a function of the training observations $\boldsymbol{X}$ (and regression observation $\boldsymbol{x}$).

The joint distribution can then be written as

$$\begin{bmatrix} f(\boldsymbol{x}) \\ \boldsymbol{y} \end{bmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} k(\boldsymbol{x},\boldsymbol{x}) & \mathbf{k}(\boldsymbol{x},\boldsymbol{X})' \\ \mathbf{k}(\boldsymbol{x},\boldsymbol{X}) & \mathbf{K}(\boldsymbol{X},\boldsymbol{X}) + \sigma_{\mathtt{n}}^2\mathbf{I} \end{bmatrix}\right)$$

The conditional distribution can then be expressed as

$$f(\boldsymbol{x})|\boldsymbol{X},\boldsymbol{y},\boldsymbol{x} \sim \mathcal{N}(\mu_{\mathtt{f}}, \sigma_{\mathtt{f}}^2)$$

where (using equalities from slide 7)

$$\begin{aligned} \mu_{\mathtt{f}} &= \mathbf{k}(\boldsymbol{x},\boldsymbol{X})'\left[\mathbf{K}(\boldsymbol{X},\boldsymbol{X}) + \sigma_{\mathtt{n}}^2\mathbf{I}\right]^{-1}\boldsymbol{y} \\ \sigma_{\mathtt{f}}^2 &= k(\boldsymbol{x},\boldsymbol{x}) - \mathbf{k}(\boldsymbol{x},\boldsymbol{X})'\left[\mathbf{K}(\boldsymbol{X},\boldsymbol{X}) + \sigma_{\mathtt{n}}^2\mathbf{I}\right]^{-1}\mathbf{k}(\boldsymbol{x},\boldsymbol{X}) \end{aligned}$$

and

$$\mathbf{k}(\boldsymbol{x},\boldsymbol{X}) = \begin{bmatrix} k(\boldsymbol{x},\boldsymbol{x}_1) \\ \vdots \\ k(\boldsymbol{x},\boldsymbol{x}_n) \end{bmatrix}, \quad \mathbf{K}(\boldsymbol{X},\boldsymbol{X}) = \begin{bmatrix} k(\boldsymbol{x}_1,\boldsymbol{x}_1) & \dots & k(\boldsymbol{x}_n,\boldsymbol{x}_1) \\ \vdots & \ddots & \vdots \\ k(\boldsymbol{x}_1,\boldsymbol{x}_n) & \dots & k(\boldsymbol{x}_n,\boldsymbol{x}_n) \end{bmatrix}$$

Basis Function Regression (Revisited)

Consider the linear model for regression

$$f(\boldsymbol{x}) = \sum_{i=1}^{n} w_i \phi(||\boldsymbol{x} - \boldsymbol{x}_i||) = \boldsymbol{w}'\boldsymbol{\phi}(\boldsymbol{x})$$

where

$$\boldsymbol{\phi}(\boldsymbol{x}) = \left[\begin{array}{ccc} \phi(||\boldsymbol{x} - \boldsymbol{x}_1||) & \ldots & \phi(||\boldsymbol{x} - \boldsymbol{x}_n||) \end{array} \right]'$$

with a Gaussian prior distribution $\boldsymbol{w} \sim \mathcal{N}(\mathbf{0}, \sigma_{\mathtt{w}}^2\mathbf{I})$ then

$$\mathcal{E}\{f(\boldsymbol{x})\} = \mathcal{E}\{\boldsymbol{w}'\}\boldsymbol{\phi}(\boldsymbol{x}) = 0$$

$$\begin{aligned} \mathcal{E}\{f(\boldsymbol{x})f(\boldsymbol{x})\} &= \boldsymbol{\phi}(\boldsymbol{x})'\mathcal{E}\{\boldsymbol{w}\boldsymbol{w}'\}\boldsymbol{\phi}(\boldsymbol{x}) \\ &= \sigma_{\mathtt{w}}^2\boldsymbol{\phi}(\boldsymbol{x})'\boldsymbol{\phi}(\boldsymbol{x}) \end{aligned}$$

This has the same form as the kernels previously seen where $\boldsymbol{\phi}(\boldsymbol{x})$ defines the feature-space.

$$\mathcal{E}\{f(\boldsymbol{x}_i)f(\boldsymbol{x}_j)\} = k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \sigma_{\mathtt{w}}^2 \left\langle \boldsymbol{\phi}(\boldsymbol{x}_i), \boldsymbol{\phi}(\boldsymbol{x}_j) \right\rangle$$

The distribution of the training outputs ($p(\boldsymbol{y}|\boldsymbol{X})$) can then be written in terms of kernel operations

$$\boldsymbol{\Sigma}_y = \sigma_{\mathtt{w}}^2\boldsymbol{\Phi}\boldsymbol{\Phi}' + \sigma_{\mathtt{n}}^2\mathbf{I} = \mathbf{K}(\boldsymbol{X}, \boldsymbol{X}) + \sigma_{\mathtt{n}}^2\mathbf{I}$$

again

$$\mathbf{K}(\boldsymbol{X}, \boldsymbol{X}) = \left[\begin{array}{ccc} k(\boldsymbol{x}_1, \boldsymbol{x}_1) & \ldots & k(\boldsymbol{x}_n, \boldsymbol{x}_1) \\ \vdots & \ddots & \vdots \\ k(\boldsymbol{x}_1, \boldsymbol{x}_n) & \ldots & k(\boldsymbol{x}_n, \boldsymbol{x}_n) \end{array} \right]$$

Covariance Functions and Kernels

The form of $\mathbf{K}(\boldsymbol{X}, \boldsymbol{X})$ is determined by the covariance function. However it has the same form as the Gram matrix used when describing kernels with SVM.

The Gram matrix associated with a covariance functions must be positive semi-definite (PSD) i.e. for a real matrix $\mathbf{K}$

$$\mathbf{v}'\mathbf{K}\mathbf{v} \geq 0$$

and these matrices will also be symmetric, $k(\boldsymbol{x}_i, \boldsymbol{x}_j) = k(\boldsymbol{x}_j, \boldsymbol{x}_i)$. It is possible to have kernels that are not PSD, but Covariance Functions must be PSD.

In the Gaussian Process literature covariance functions are split into two broad groups:

- stationary: are a function of $\boldsymbol{x}_i - \boldsymbol{x}_j$. These are invariant to translations in the input space. The various forms of basis interpolation will fall under this category.
- non-stationary: these are functions which do vary with translation.

  The most important sub-set of these covariance functions are dot-product covariance functions. These are only functions of $\langle \boldsymbol{x}_i, \boldsymbol{x}_j \rangle$.

  – linear: $k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \langle \boldsymbol{x}_i, \boldsymbol{x}_j \rangle$
  – polynomial, order d: $k(\boldsymbol{x}_i, \boldsymbol{x}_j) = (\langle \boldsymbol{x}_i, \boldsymbol{x}_j \rangle + 1)^d$

Squared Exponential Covariance Function

One standard stationary covariance function is given by

$$k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \alpha \exp\left(-\frac{||\boldsymbol{x}_i - \boldsymbol{x}_j||^2}{2\sigma_{\mathtt{l}}^2}\right)$$

This has free parameters (or hyper-parameters)

- length scale - $\sigma_{\mathtt{l}}$
- signal variance - α

This is often used as the covariance function over the systematic prediction function, $f(\boldsymbol{x})$. The covariance function for the observation would be

$$k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \alpha \exp\left(\frac{-||\boldsymbol{x}_i - \boldsymbol{x}_j||^2}{2\sigma_{\mathtt{l}}^2}\right)$$

There will normally be some prediction noise. In this case the covariance function will become

$$k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \alpha \exp\left(-\frac{||\boldsymbol{x}_i - \boldsymbol{x}_j||^2}{2\sigma_{\mathtt{l}}^2}\right) + \sigma_{\mathtt{n}}^2 \delta_{ij}$$

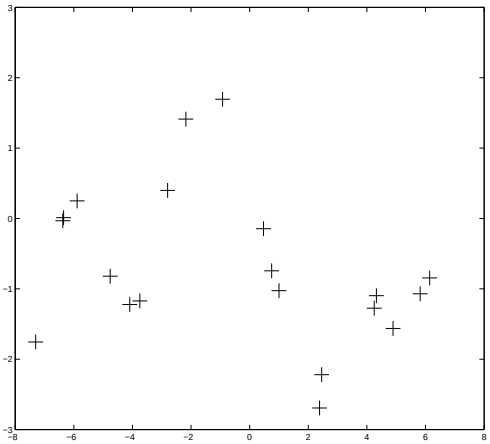
where

$$\delta_{ij} = \begin{cases} 1, & i = j \\ 0, & \text{otherwise} \end{cases}$$

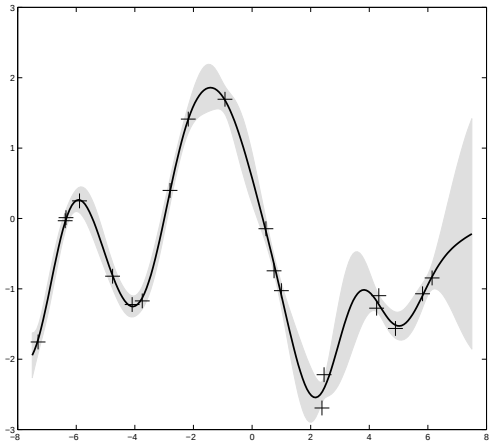
Example Gaussian Process

Consider a covariance function of the form

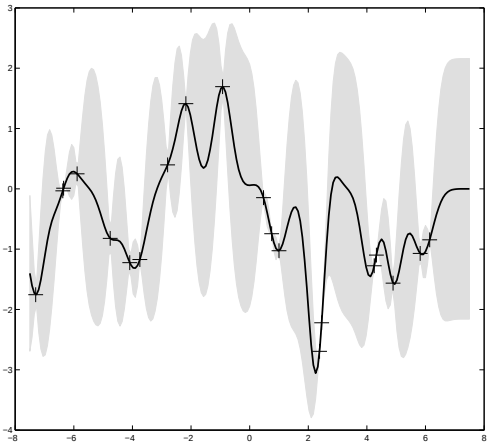
$$k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \alpha \exp\left(-\frac{||\boldsymbol{x}_i - \boldsymbol{x}_j||^2}{2\sigma_{\mathtt{l}}^2}\right) + \sigma_{\mathtt{n}}^2 \delta_{ij}$$



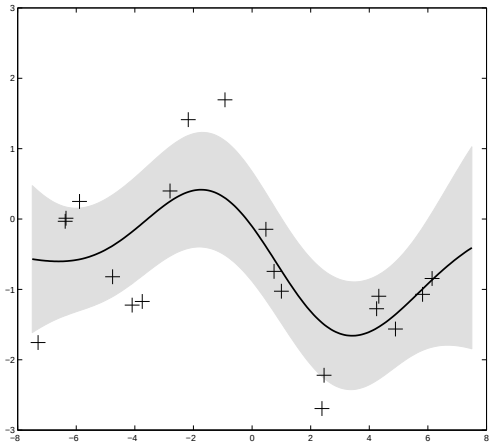
Points (drawn from right)



$\sigma_{\mathtt{l}} = 1.0, \alpha = 1.0, \sigma_{\mathtt{n}} = 0.1$



$\sigma_{\mathtt{l}} = 0.3, \alpha = 1.08, \sigma_{\mathtt{n}} = 5.0 \times 10^{-5}$



$\sigma_{\mathtt{l}} = 3.0, \alpha = 1.16, \sigma_{\mathtt{n}} = 0.89$

The grey region shows the +/- 2 standard deviations (95% confidence) - $f(\boldsymbol{x})$ only (noise std, $\sigma_{\mathtt{n}}$ subtracted)

Gaussian Process Practicalities

It is interesting to note that it is possible to write the mean of the prediction

$$\mu_{\mathtt{f}} = \sum_{i=1}^{n} \alpha_i k(\boldsymbol{x}, \boldsymbol{x}_i)$$

where

$$\boldsymbol{\alpha} = \left[\mathbf{K}(\boldsymbol{X}, \boldsymbol{X}) + \sigma_{\mathtt{n}}^2 \mathbf{I}\right]^{-1} \boldsymbol{y}$$

In general all values of α_i will be non-zero.

Thus regression/prediction is expressed in a non-sparse combination of the kernel between the "test" observation and all the training points. This can be contrasted with the SVM where the decision boundary is only a function of the support vectors (thus it is highly sparse).

In the same fashion as SVM decision boundary Gaussian process regression is limited to the space spanned within the feature-space ($\boldsymbol{\phi}(\boldsymbol{x})$) of all the training points.

One of the major problems with implementing Gaussian Processes is the computational cost. This is dominated by the need to invert $\mathbf{K}(\boldsymbol{X}, \boldsymbol{X}) + \sigma_{\mathtt{n}}^2 \mathbf{I}$. This has dimensionality of the number of training examples. For large training sets a number of approximate approaches can be used.

Learning Hyper-parameters

So far it has been assumed that the free parameters, the hyper-parameters of the covariance function, $\boldsymbol{\theta}$, are known. It would be preferable to be able to train these.

One approach to do this is to maximise the marginal likelihood. This is (remembering the Bayesian interpretation of linear regression and assuming zero mean)

$$p(\boldsymbol{y}|\boldsymbol{X}, \boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{y}; \mathbf{0}, \mathbf{K}_\theta(\boldsymbol{X}, \boldsymbol{X}) + \sigma_{\mathrm{n}}^2 \mathbf{I})$$

where $\boldsymbol{\theta}$ are the hyper-parameters.

It is possible to write

$$\log(p(\boldsymbol{y}|\boldsymbol{X}, \boldsymbol{\theta})) = -\frac{1}{2}\boldsymbol{y}'\boldsymbol{\Sigma}_y^{-1}\boldsymbol{y} - \frac{1}{2}\log(|\boldsymbol{\Sigma}_y|) - \frac{n}{2}\log(2\pi)$$

where

$$\boldsymbol{\Sigma}_y = \mathbf{K}_\theta(\boldsymbol{X}, \boldsymbol{X}) + \sigma_{\mathrm{n}}^2 \mathbf{I}$$

This expression may be considered to have two parts

- data fit: $-\frac{1}{2}\boldsymbol{y}'\boldsymbol{\Sigma}_y^{-1}\boldsymbol{y}$
- complexity penalty: $-\frac{1}{2}\log(|\boldsymbol{\Sigma}_y|)$

(the last term is a constant). Using this expression it is also possible to estimate the noise variance, σ_{n}^2 as well as $\boldsymbol{\theta}$.

Hyper-parameter Optimisation

Matrix Calculus (Reference)

We will need the gradient which will make use of

$$\begin{aligned}\frac{\partial}{\partial\theta}\mathbf{K}^{-1} &= -\mathbf{K}^{-1}\frac{\partial\mathbf{K}}{\partial\theta}\mathbf{K}^{-1}\\ \frac{\partial}{\partial\theta}\log(|\mathbf{K}|) &= \text{tr}\left(\mathbf{K}^{-1}\frac{\partial\mathbf{K}}{\partial\theta}\right)\end{aligned}$$

where

$$\text{tr}(\mathbf{K}) = \sum_{i=1}^{n} k_{ii}$$

assuming that $\mathbf{K}$ is an $n \times n$ matrix.

Hyper-parameter Derivatives

We now need the partial derivatives of the marginal likelihood with respect to the hyper-parameters. Using the matrix calculus equalities

$$\begin{aligned}\frac{\partial}{\partial\theta_j}\log(p(\boldsymbol{y}|\boldsymbol{X},\boldsymbol{\theta})) &= \frac{1}{2}\boldsymbol{y}'\mathbf{K}^{-1}\frac{\partial\mathbf{K}}{\partial\theta_j}\mathbf{K}^{-1}\boldsymbol{y} - \frac{1}{2}\text{tr}\left(\mathbf{K}^{-1}\frac{\partial\mathbf{K}}{\partial\theta_j}\right)\\ &= \frac{1}{2}\text{tr}\left((\mathbf{K}^{-1}\boldsymbol{y}\boldsymbol{y}'\mathbf{K}'^{-1} - \mathbf{K}^{-1})\frac{\partial\mathbf{K}}{\partial\theta_j}\right)\end{aligned}$$

(using $\mathbf{K}$ as a shorthand for $\mathbf{K}_\theta(\boldsymbol{X},\boldsymbol{X})$ and the equality $\boldsymbol{y}'\mathbf{A}\boldsymbol{y} = \text{tr}(\boldsymbol{y}\boldsymbol{y}'\mathbf{A})$). The computational cost of this is dominated by the inversion of $\mathbf{K}$ which has cost $\mathcal{O}(n^3)$

It is not possible to guarantee that the marginal likelihood doesn't suffer from multiple local maxima. Each of these will correspond to different interpretations of the data.

Classification with Regression Models

The examples of Gaussian processes given have been examined for regression. They may also be used for classification.

To use the output from Gaussian process regression, requires mapping from points in the range $-\infty$ to ∞ into "probabilities" in the range 0 to 1. There are a number of ways that this can be done. Initially considering just the binary case:

- logistic regression: here (considering linear regression)

$$P(y = +1|\boldsymbol{x}, \hat{\boldsymbol{w}}) = \frac{1}{1 + \exp(-\hat{\boldsymbol{w}}'\boldsymbol{x})}$$

- probit regression: again considering linear regression

$$P(y = +1|\boldsymbol{x}, \hat{\boldsymbol{w}}) = \int_{-\infty}^{\hat{\boldsymbol{w}}'\boldsymbol{x}} \mathcal{N}(z; 0, 1)dz$$

Using the sum to one constraint

$$P(y = -1|\boldsymbol{x}, \hat{\boldsymbol{w}}) = 1 - P(y = +1|\boldsymbol{x}, \hat{\boldsymbol{w}})$$

These appear to be noise free - in the sense that they appear in the same form as "noise-less" regression $y = f(\boldsymbol{x})$. It can be shown that probit regression is related to a 1/0 probability

$$P(y = +1|\boldsymbol{x}, \hat{\boldsymbol{w}}) = \begin{cases} 1, & \hat{\boldsymbol{w}}'\boldsymbol{x} + \epsilon \geq 0 \\ 0, & \text{otherwise} \end{cases}$$

with Gaussian noise (see examples paper).

How to apply the same concepts to Gaussian Processes?

Bayesian Classification

In the same fashion as Bayesian regression Bayesian classification involves integrating (marginalising) over the model parameter distribution.
Before approaching more general Gaussian processes, consider the simple case of $f(\boldsymbol{x}) = \boldsymbol{w}'\boldsymbol{x}$, then classification is based on

$$P(y = +1|\boldsymbol{x}, \boldsymbol{X}, \boldsymbol{y}) = \int P(y = +1|\boldsymbol{x}, \boldsymbol{w})p(\boldsymbol{w}|\boldsymbol{X}, \boldsymbol{y})d\boldsymbol{w}$$

In the same fashion as prediction we need $p(\boldsymbol{w}|\boldsymbol{X}, \boldsymbol{y})$.

Again using the Gaussian prior for the weights

$$p(\boldsymbol{w}|\boldsymbol{X}, \boldsymbol{y}) \propto p(\boldsymbol{w})P(\boldsymbol{y}|\boldsymbol{X}, \boldsymbol{w})$$

The targets are conditionally independent, so

$$\log(p(\boldsymbol{w}|\boldsymbol{X}, \boldsymbol{y})) \propto -\frac{1}{\sigma_{\mathtt{w}}^2}\boldsymbol{w}'\boldsymbol{w} + \sum_{i=1}^{n} \log(P(y_i|\boldsymbol{x}_i, \boldsymbol{w}))$$

Unlike the regression case this is not Gaussian distributed. The posterior of the weights given the training data does not have a simple analytical solution.

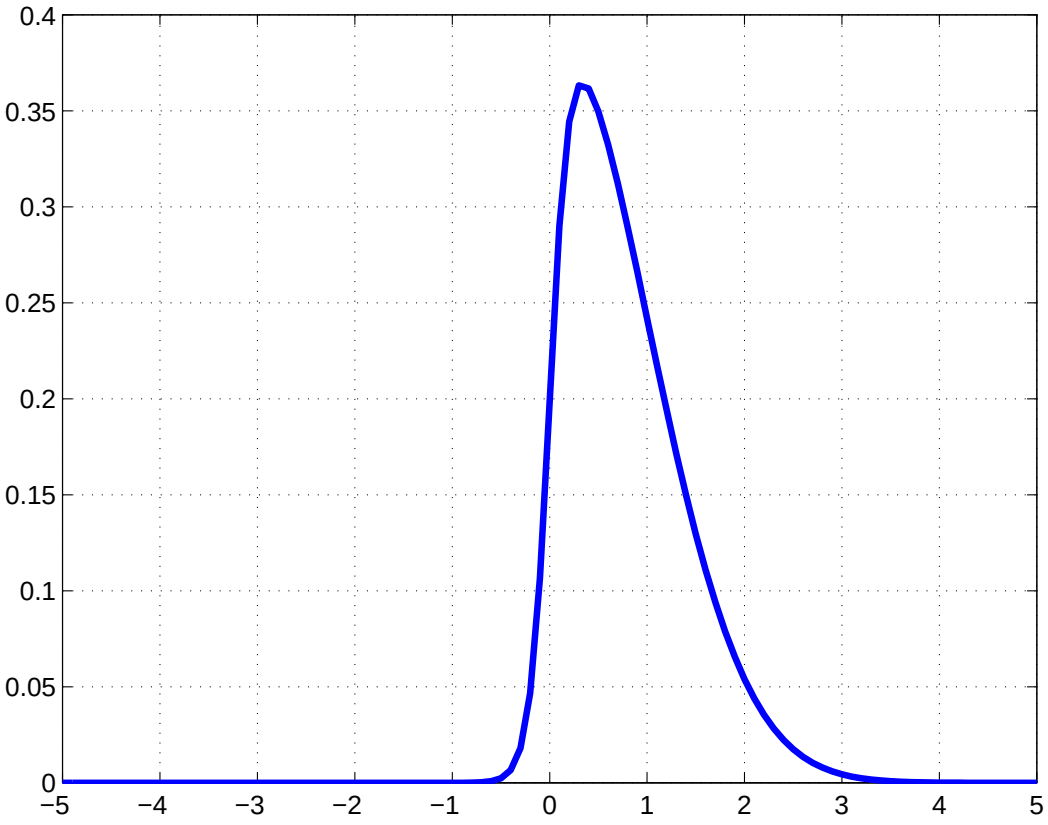
Posterior Distribution Example

Taking a simple example of what the posterior looks like for $f(x) = wx$. Here

- weight prior is Gaussian $p(w) = \mathcal{N}(w; 0, 1)$
- single training observation $y_1 = 1$, $x_1 = 10$

Posterior (using logistic regression) is given by

$$p(w|y_1, x_1) \propto \mathcal{N}(w; 0, 1) \left(\frac{1}{1 + \exp(-10w)} \right)$$



Weight posterior is non-Gaussian

Gaussian Process Classification

For the more general classification, the function $f(\boldsymbol{x})$ will be modelled using a Gaussian process. For the training data the distribution of the vector of training functions $\mathbf{f}(.)$

$$\mathbf{f}(.)|\boldsymbol{X} = \begin{bmatrix} f(\boldsymbol{x}_1) \\ \vdots \\ f(\boldsymbol{x}_n) \end{bmatrix} \sim \mathcal{N}(\mathbf{0}, \mathbf{K}(\boldsymbol{X}, \boldsymbol{X}))$$

In contrast to the regression case even if $\mathbf{f}(.)|\boldsymbol{X}$ is Gaussian distributed , the output targets, $\boldsymbol{y}$ will not be Gaussian distributed.

In a similar fashion to classification with linear prediction, the outputs are conditionally independent given the systematic prediction

$$P(\boldsymbol{y}|\mathbf{f}(.)) = \prod_{i=1}^{n} P(y_i|f(\boldsymbol{x}_i))$$

If logistic regression is used then (note targets 1 and 0)

$$\begin{aligned} \log(P(\boldsymbol{y}|\mathbf{f}(.))) = \sum_{i=1}^{n} \Bigg[& y_i \log\left(\frac{1}{1+\exp(-f(\boldsymbol{x}_i))}\right) \\ & +(1-y_i)\log\left(1-\frac{1}{1+\exp(-f(\boldsymbol{x}_i))}\right)\Bigg] \end{aligned}$$

The non-linear relationship between the targets and the systematic prediction is again clear. The classification posterior is then given by

$$P(y|\boldsymbol{X}, \boldsymbol{x}, \boldsymbol{y}) = \int P(y|f(\boldsymbol{x}))p(f(\boldsymbol{x})|\boldsymbol{X}, \boldsymbol{y}, \boldsymbol{x})df(\boldsymbol{x})$$

Gaussian Process Classification (cont)

Now considering the general process procedure. The distributions over the functions is again assumed to be Gaussian. Binary classification proceeds in two steps

- function posterior distribution: first the distribution over the functions given the training data and the observation is required.

  $$p(f(\boldsymbol{x})|\boldsymbol{X},\boldsymbol{y},\boldsymbol{x}) = \int p(f(\boldsymbol{x})|\boldsymbol{X},\boldsymbol{x},\mathbf{f}(.))p(\mathbf{f}(.)|\boldsymbol{X},\boldsymbol{y})d\mathbf{f}(.)$$

  As the relationship between the function and the target is no longer linear

  $$p(\mathbf{f}(.)|\boldsymbol{X},\boldsymbol{y}) \propto P(\boldsymbol{y}|\mathbf{f}(.))p(\mathbf{f}(.)|\boldsymbol{X})$$

  will not be Gaussian distributed even if the function (as here) is modelled using a Gaussian process.

- marginalise over distribution: consider logistic regression

  $$P(y=+1|\boldsymbol{x},\boldsymbol{X},\boldsymbol{y}) = \int \left(\frac{1}{1+\exp(-f(\boldsymbol{x}))}\right) p(f(\boldsymbol{x})|\boldsymbol{X},\boldsymbol{y},\boldsymbol{x})df(\boldsymbol{x})$$

Both these integrals are, in general, not analytical solvable:

- the marginalisation over the distribution is possible using standard numerical techniques as it is 1-dimensional integral.
- finding the function posterior is more complicated.

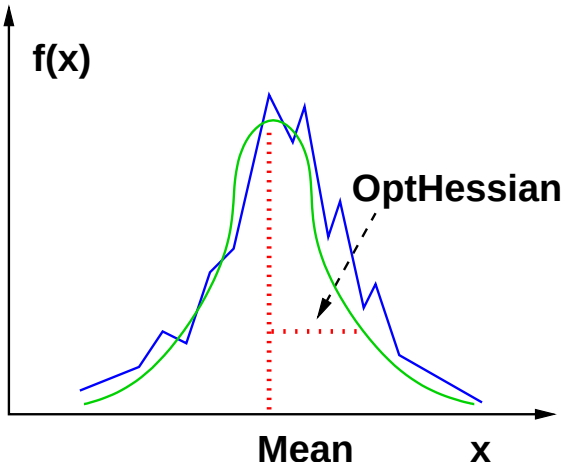
Laplace's Approximation

It is necessary to find the function posterior distribution

$$p(f(\boldsymbol{x})|\boldsymbol{X}, \boldsymbol{y}, \boldsymbol{x}) = \int p(f(\boldsymbol{x})|\boldsymbol{X}, \boldsymbol{x}, \mathbf{f}(.))p(\mathbf{f}(.)|\boldsymbol{X}, \boldsymbol{y})d\mathbf{f}(.)$$

As the function, $f(\boldsymbol{x})$ is modelled using a Gaussian process then $p(f(\boldsymbol{x})|\boldsymbol{X}, \boldsymbol{x}, \mathbf{f}(.))$ is Gaussian distributed.

A general approximation technique to modelling posterior distributions is to use Laplace's approximation.



Here a Gaussian distribution is used:

$$p(\mathbf{f}(.)|\boldsymbol{X}, \boldsymbol{y}) \approx q(\mathbf{f}(.)|\boldsymbol{X}, \boldsymbol{y}) = \mathcal{N}(\mathbf{f}(.); \boldsymbol{\mu}_{\mathtt{f}}, \boldsymbol{\Sigma}_{\mathtt{f}})$$

where

$$\begin{aligned} \boldsymbol{\mu}_{\mathtt{f}} &= \arg\max_{\mathbf{f}} \{p(\mathbf{f}(.)|\boldsymbol{X}, \boldsymbol{y})\} \\ \boldsymbol{\Sigma}_{\mathtt{f}}^{-1} &= -\boldsymbol{\nabla}^2 \log(p(\mathbf{f}(.)|\boldsymbol{X}, \boldsymbol{y}))\big|_{\mathbf{f}(.)=\boldsymbol{\mu}_{\mathtt{f}}} \end{aligned}$$

($\boldsymbol{\Sigma}_{\mathtt{f}}^{-1}$ is the Hessian evaluated at $\boldsymbol{\mu}_{\mathtt{f}}$)

Laplace's Approximation (cont)

Applying Laplace's approximation to Gaussian processes

$$p(\mathbf{f}(.)|\boldsymbol{X},\boldsymbol{y}) = \frac{P(\boldsymbol{y}|\mathbf{f}(.))p(\mathbf{f}(.)|\boldsymbol{X})}{P(\boldsymbol{y}|\boldsymbol{X})}$$

As the denominator is not a function of $\mathbf{f}(.)$ then need to find

$$\max_{\mathbf{f}(.)} \{\log(P(\boldsymbol{y}|\mathbf{f}(.))) + \log(p(\mathbf{f}(.)|\boldsymbol{X}))\} = \max_{\mathbf{f}(.)} \{\mathcal{F}(\mathbf{f}(.))\}$$

This may be expressed as

$$\max_{\mathbf{f}(.)} \left\{\log(P(\boldsymbol{y}|\mathbf{f}(.))) - \frac{1}{2}\mathbf{f}(.)\mathbf{K}^{-1}\mathbf{f}(.) - \frac{1}{2}\log(|\mathbf{K}|)\right\}$$

($\mathbf{K} = \mathbf{K}(\boldsymbol{X},\boldsymbol{X})$). Differentiating this yields

$$\begin{aligned}\boldsymbol{\nabla}\mathcal{F}(\mathbf{f}(.)) &= \boldsymbol{\nabla}\log(P(\boldsymbol{y}|\mathbf{f}(.))) - \mathbf{K}^{-1}\mathbf{f}(.)\\ \boldsymbol{\nabla}^2\mathcal{F}(\mathbf{f}(.)) &= \boldsymbol{\nabla}^2\log(P(\boldsymbol{y}|\mathbf{f}(.))) - \mathbf{K}^{-1}\end{aligned}$$

Note each of the observations are assumed independent. Taking the example of logistic regression it is simple to show (and used for logistic regression/classification training previously)

$$\frac{\partial}{\partial f(\boldsymbol{x}_i)}\log(P(y_i|f(\boldsymbol{x}_i))) = y_i - \frac{1}{1+\exp(-f(\boldsymbol{x}_i))}$$

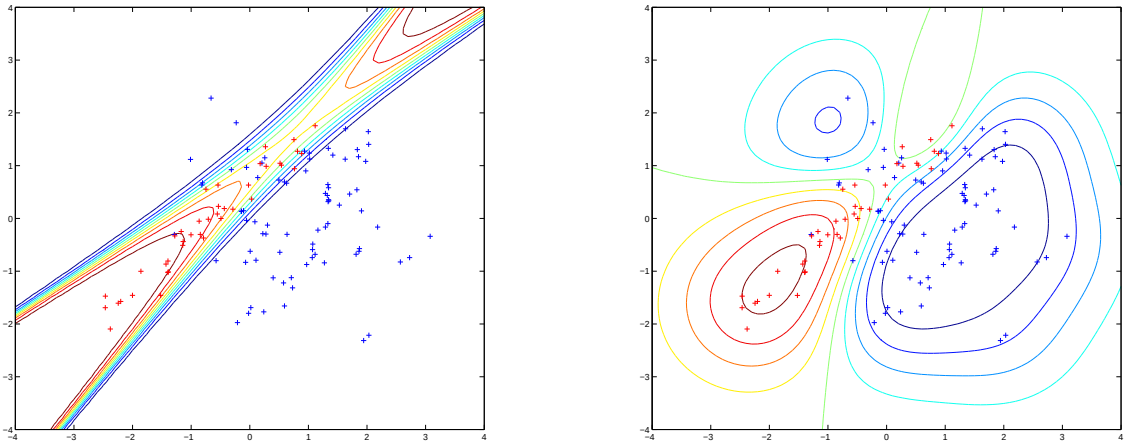
The maximum can be simply found using Newton's method for example.

Classification Example

Data generated from two Gaussian Distributions

$$\boldsymbol{x}^{(1)} \sim \mathcal{N}\left(\begin{bmatrix} 0.75 \\ 0 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}\right) \quad \boldsymbol{x}^{(2)} \sim \mathcal{N}\left(\begin{bmatrix} -0.75 \\ 0 \end{bmatrix} \begin{bmatrix} 1 & 0.95 \\ 0.95 & 1 \end{bmatrix}\right)$$

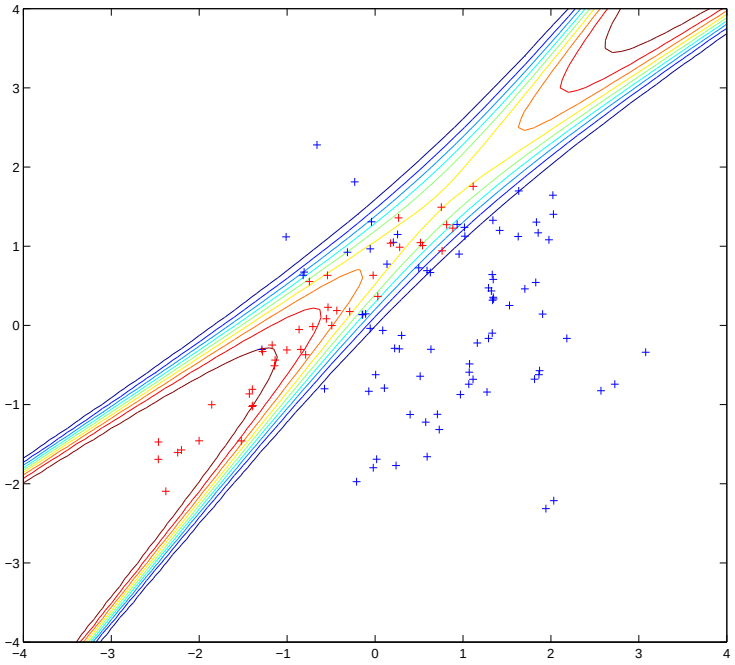
Given the two class-conditional PDFs (and assuming equal priors) it is possible to define decision boundaries/class posteriors (left diagram).

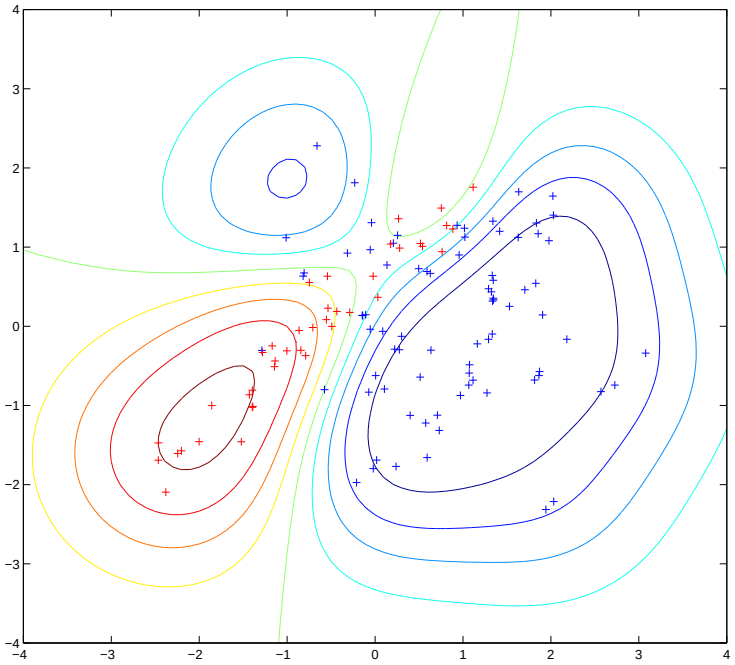


The right diagram shows an application of Laplace's approximation with a squared exponential covariance functions with

$$\sigma_1 = 1, \quad \alpha = 1$$

- posteriors differ, but similar in high data density regions
- tend to posteriors of 0.5 as we move away from the data

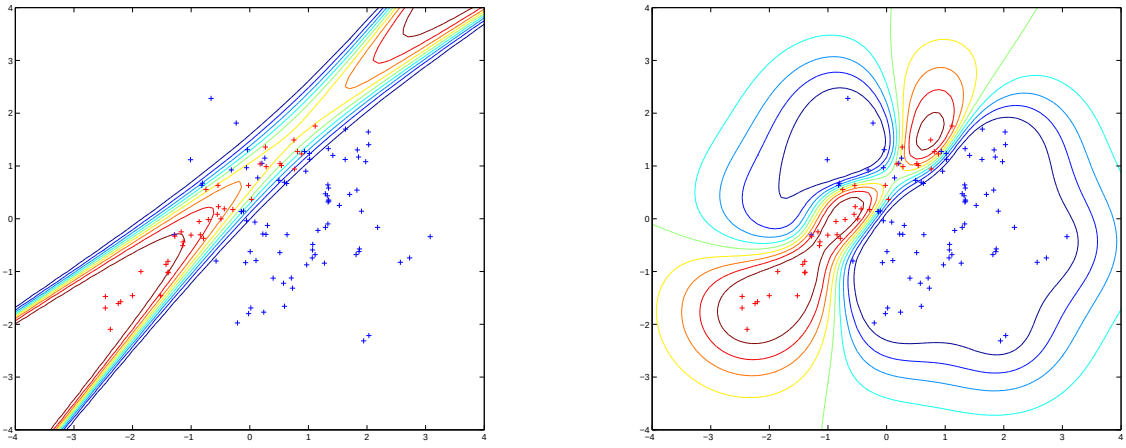

4
3
2
1
0
−1
−2
−3
−4
−4
−3
−2
−1
0
1
2
3
4


4
3
2
1
0
−1
−2
−3
−4
−4
−3
−2
−1
0
1
2
3
4

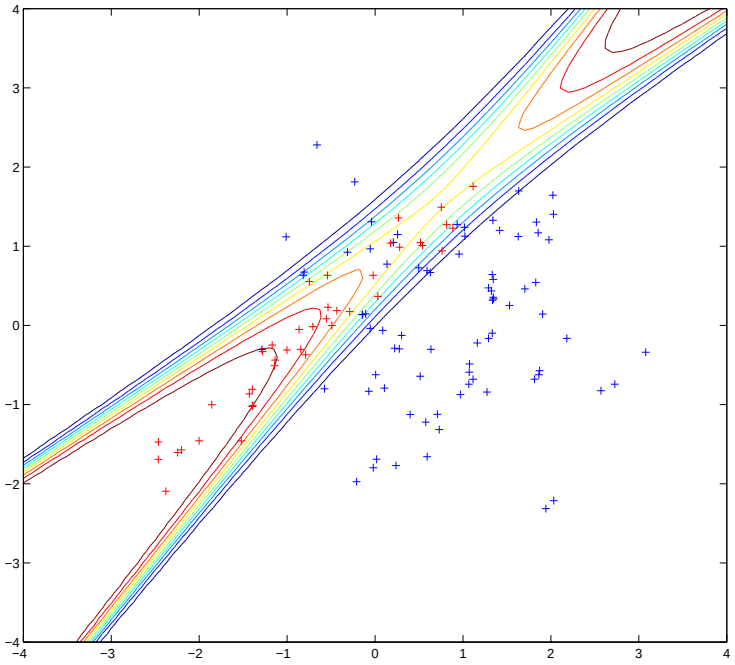
Classification Hyper-parameter Estimation

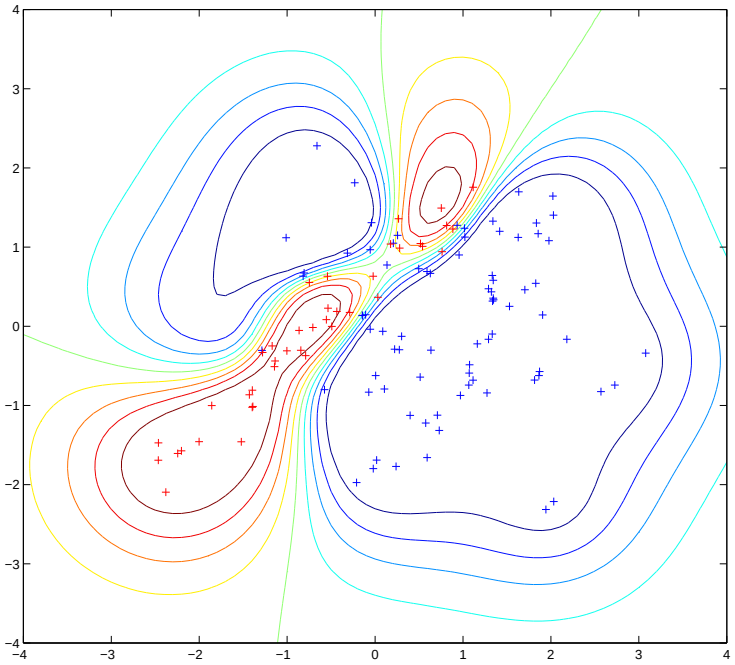
Rather than using the arbitrary hyper-parameters, these can be trained by maximising the likelihood of the outputs (using the Laplace's approximation). this yields

$$\sigma_1 = 1.280, \quad \alpha = 14.440$$



- length scale moves only a small amount
- signal variance increased - more "extreme" probabilities


4
3
2
1
0
−1
−2
−3
−4
−4
−3
−2
−1
0
1
2
3
4


4
3
2
1
0
−1
−2
−3
−4
−4
−3
−2
−1
0
1
2
3
4

Summary

The last two lectures have examined the use of Gaussian Processes for prediction/regression and classification. In particular:

- linear/basis function regression
- Bayesian regression
- definition of a Gaussian process
- covariance functions and kernels
- Gaussian processes for regression
- learning hyper-parameters
- Gaussian processes for classification
- Laplace's approximation