Probabilistic Approaches for Pattern Recognition

Anil Rao
(Based on slides from Andre Marquand)

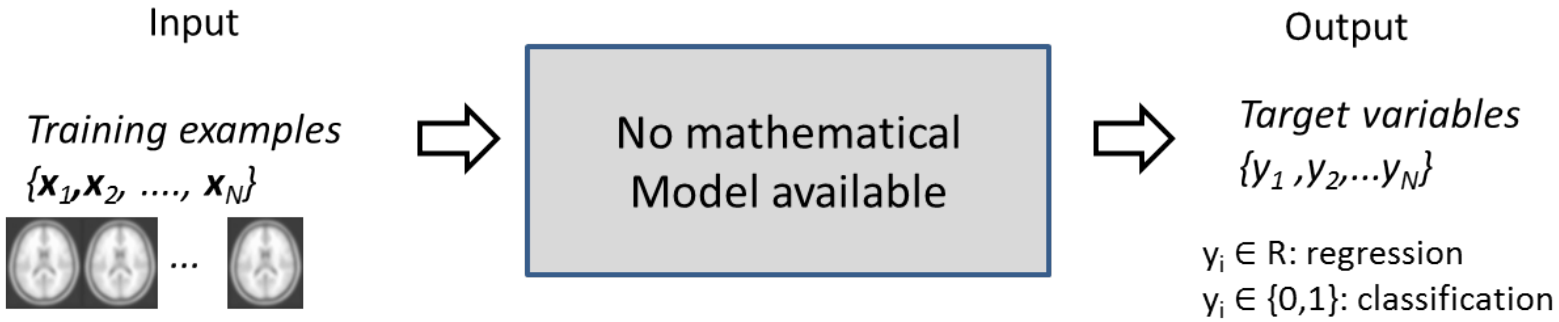
May 30, 2017

Outline

Outline

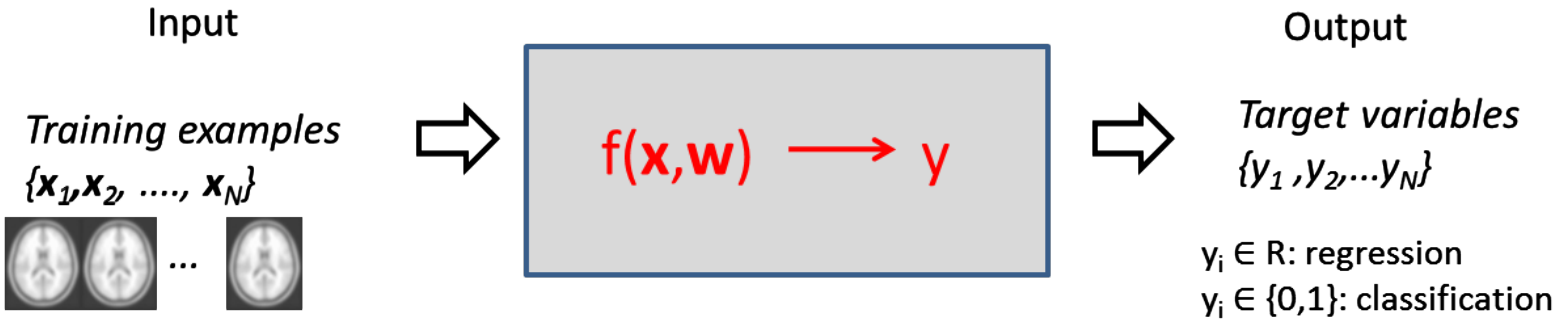
Overview of PR in Neuroimaging

PR involves learning a mapping between input and output:



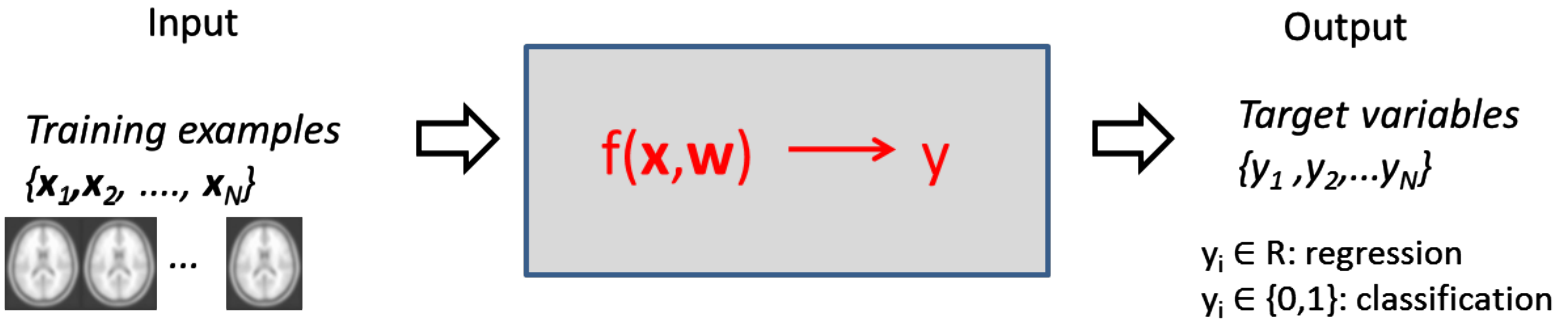
Overview of PR in Neuroimaging

PR involves learning a mapping between input and output:



Overview of PR in Neuroimaging

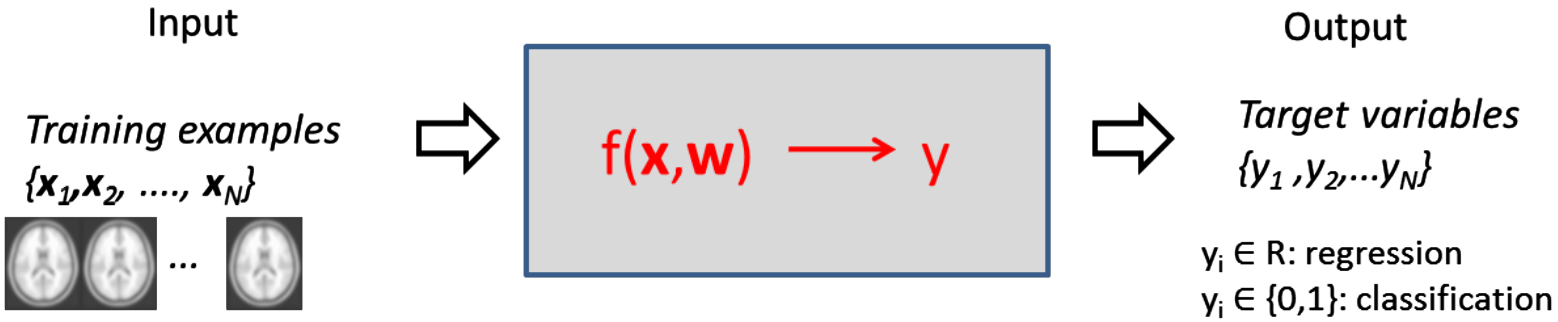
PR involves learning a mapping between input and output:



PR techniques hold two main advantages over conventional univariate analytic methods:

Overview of PR in Neuroimaging

PR involves learning a mapping between input and output:

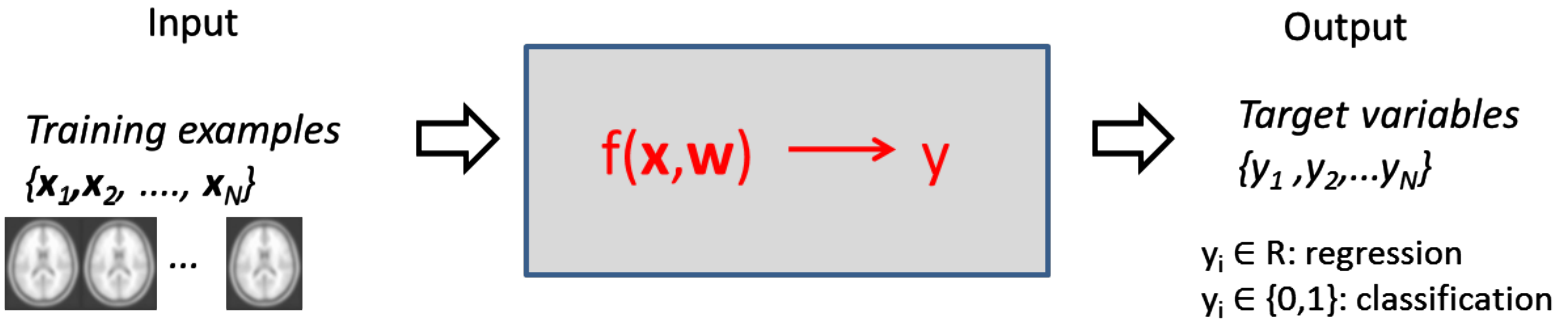


PR techniques hold two main advantages over conventional univariate analytic methods:

1. They can make **predictions** at the level of single subjects

Overview of PR in Neuroimaging

PR involves learning a mapping between input and output:



PR techniques hold two main advantages over conventional univariate analytic methods:

1. They can make **predictions** at the level of single subjects
2. They can make use of correlations between brain regions (i.e. they are **multivariate**)

Approaches to Pattern Recognition

There are many different algorithms used for PR, which often overlap with conventional statistical methods

Algorithms

- Neural Networks
- Random Forests / Decision Trees
- LASSO / Elastic Net
- Linear Discriminant Analysis
- Kernel methods (e.g. Support Vector Machines, Gaussian Processes, Relevance Vector Machines)

Some algorithms are inherently probabilistic (others aren't)
Under the probabilistic approach we use probability distributions to model quantities of interest

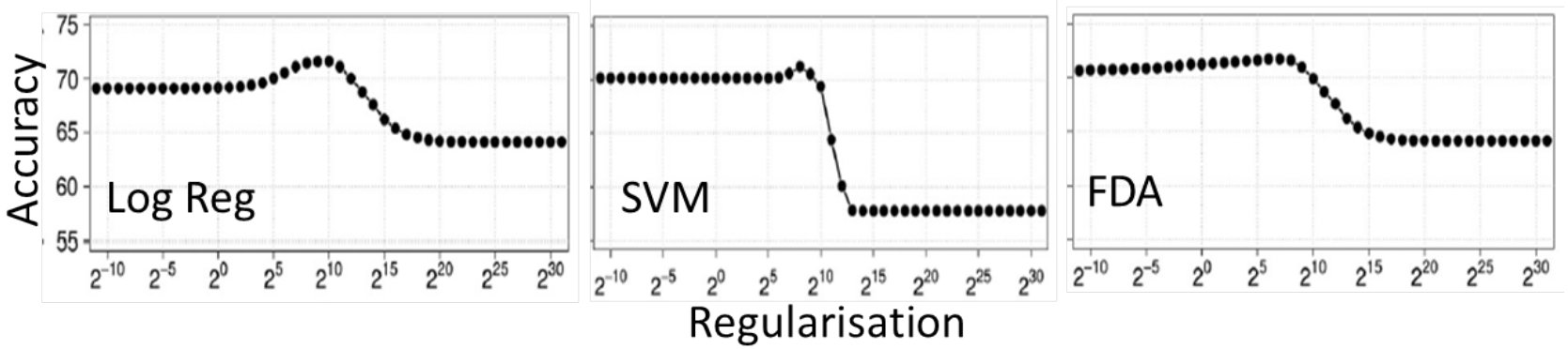
Pattern Recognition Algorithms

Pattern Recognition Algorithms

- Neuroimaging applications most often employ the binary support vector machine (SVM) classifier

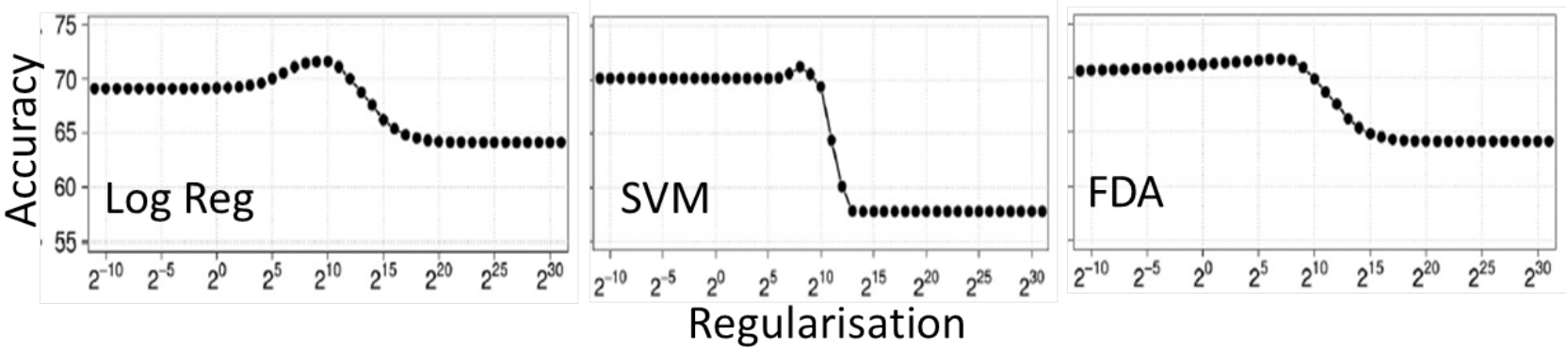
Pattern Recognition Algorithms

- Neuroimaging applications most often employ the binary support vector machine (SVM) classifier
- However, for binary classification predictive performance of most algorithms is similar (Rasmussen et al., 2011)



Pattern Recognition Algorithms

- Neuroimaging applications most often employ the binary support vector machine (SVM) classifier
- However, for binary classification predictive performance of most algorithms is similar (Rasmussen et al., 2011)



- Other factors are more important than accuracy in deciding which classifier is best suited to each application
- One example is whether the approach provides probabilistic class predictions

Outline

Probability Theory

- $p(X)$ is the *marginal* probability of X
- $p(X, Y)$ is the *joint* probability of X and Y
- $p(X|Y)$ is the *conditional* probability of X given Y

Rules

- $0 < p(X) < 1$
- p(sure thing) $= 1$
- probabilities must sum to one: $\sum_X p(X) = 1$
- Product rule: $p(X, Y) = p(X|Y)p(Y) = p(Y|X)p(X)$
- Sum rule: $p(X) = \sum_Y p(X, Y)$

Bayes rule is derived from the product rule

$$p(X|Y) = \frac{p(Y|X)p(X)}{p(Y)} \qquad \text{posterior} = \frac{\text{likelihood} \times \text{prior}}{\text{evidence}}$$

Probabilistic (Supervised) Learning

Notation

- We have with a dataset consisting of input/output pairs:

$$
\begin{aligned}
D &= \{\mathbf{x}_i, \mathbf{y}_i\}_{i=1}^n \\
\mathbf{X} &= [\mathbf{x}_1, ..., \mathbf{x}_n]^T \\
\mathbf{y} &= [y_1, ..., y_n]^T \quad \text{binary/regression} \\
\mathbf{Y} &= [\mathbf{y}_1^T, ..., \mathbf{y}_n^T] \quad \text{multi-class}
\end{aligned}
$$

$$
\begin{aligned}
\mathbf{w} &= [\mathbf{w}_1, ..., \mathbf{w}_C]^T \quad \text{parameters (weights)} \\
\sigma &= [\sigma_1, ..., \sigma_q]^T \quad \text{likelihood hyperparameters} \\
\theta &= [\theta_1, ..., \theta_p]^T \quad \text{prior hyperparameters}
\end{aligned}
$$

Probabilistic Learning continued

- To define a probabilistic model, we start with choosing the **likelihood** function which describes how the data were produced

$$p(\mathit{data}|\mathit{parameters}) = p(\mathbf{y}|\mathbf{w}, \mathbf{X}, \sigma)$$

Probabilistic Learning continued

- To define a probabilistic model, we start with choosing the **likelihood** function which describes how the data were produced

$$p(\textit{data}|\textit{parameters}) = p(\mathbf{y}|\mathbf{w}, \mathbf{X}, \sigma)$$

Many possible choices depending on our problem eg. if we are doing regression or classification.

Probabilistic Learning continued

- To define a probabilistic model, we start with choosing the **likelihood** function which describes how the data were produced

$$p(\textit{data}|\textit{parameters}) = p(\mathbf{y}|\mathbf{w}, \mathbf{X}, \sigma)$$

Many possible choices depending on our problem eg. if we are doing regression or classification.

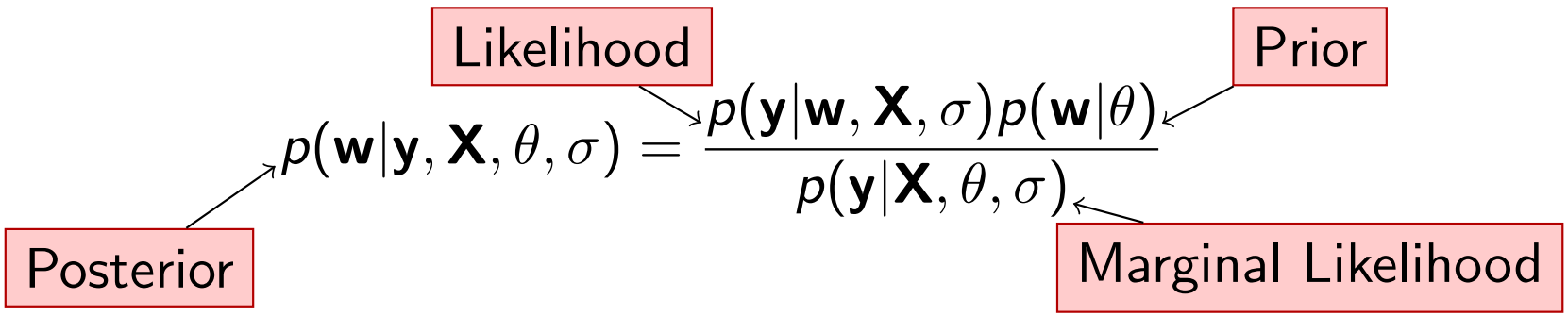
- We also specify our **prior** beliefs about the weight vector

$$p(\textit{parameters}|\textit{model}) = p(\mathbf{w}|\theta)$$

You can think of this as similar to regularisation in non-probabilistic approaches

Probabilistic Learning continued

- Inference then amounts to computing the posterior distribution (Bayes rule)



- Gives a **distribution** for the weight vector **w** given the data
  We then can use this to perform predictions

- The Marginal Likelihood enables us to perform model selection and choose the optimum values for the hyperparameters θ, σ.

Model Selection

- The marginal likelihood (evidence) plays an important role in probabilistic modeling

$$p(\mathbf{y}|\mathbf{X}, \theta, \sigma) = \int p(\mathbf{y}|\mathbf{X}, \mathbf{w}, \sigma) p(\mathbf{w}|\theta) d\mathbf{w}$$

Model Selection

- The marginal likelihood (evidence) plays an important role in probabilistic modeling

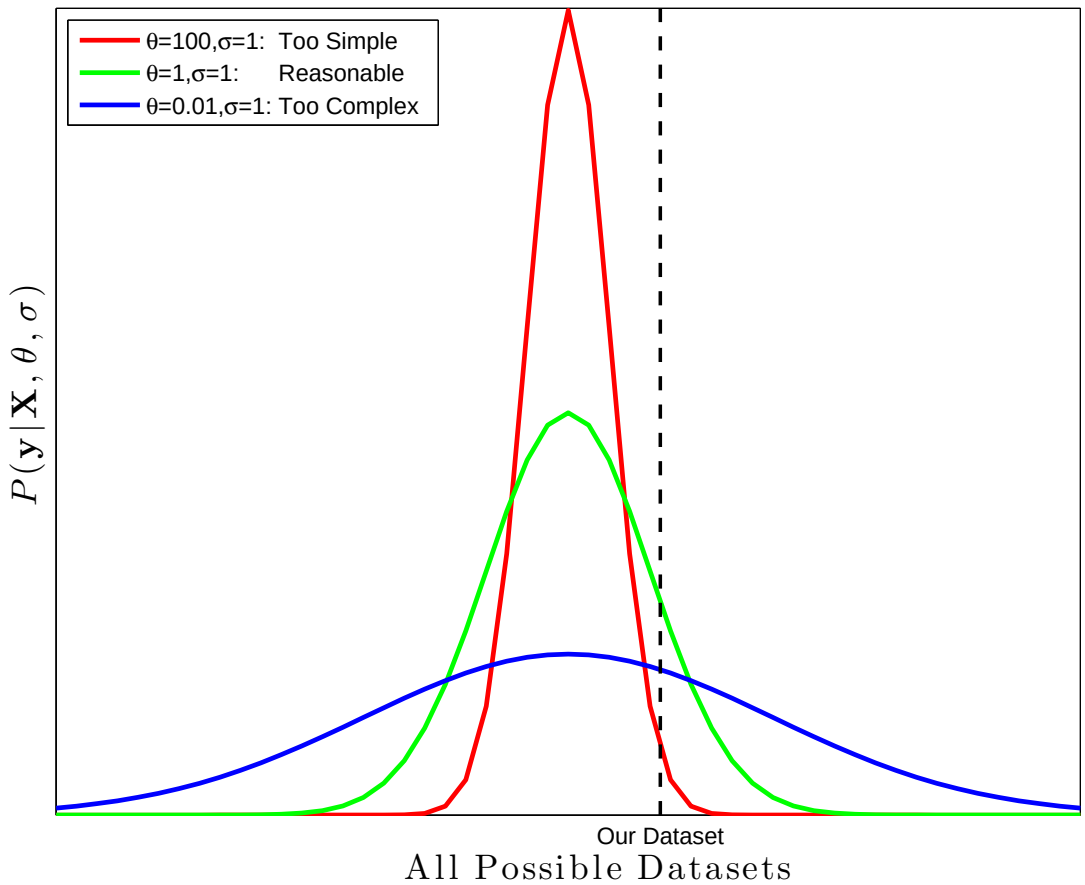
$$p(\mathbf{y}|\mathbf{X}, \theta, \sigma) = \int p(\mathbf{y}|\mathbf{X}, \mathbf{w}, \sigma) p(\mathbf{w}|\theta) d\mathbf{w}$$

  It embodies a tradeoff between data fit and model complexity and can be used for:

- deciding which of several competing models is most probable
- automatic optimisation of hyperparameters θ, σ by evidence maximisation

Model Selection

- Choosing optimum values for θ, σ



Outline

Decision Theory

In probabilistic models, we commonly divide the learning process into two phases:

1. **Inference**: computing the posterior distributions
2. **Decision**: make a prediction/decision based on the posterior

- Decision theory concerns the second step (e.g. given the class probabilities, should we choose treatment A or B?)
- This framework is highly flexible: e.g. we can accommodate asymmetric misclassification costs where a false negative may be costly than a false positive (medical applications)
- In contrast many approaches combine these phases and learn a function that directly maps inputs ($\mathbf{x}$) onto class labels (y). This is called a *discriminant function* approach (e.g. SVM)

Decision Theory

- We can formalise the measurement of model performance using some "loss function" $\mathcal{L}(y, f(\mathbf{x}))$
- There are many different loss functions for classification (e.g. classification error) and regression (e.g. MSE)
- The expected generalizability is then given by its "Risk":

$$\mathcal{R}[f] = \int \mathcal{L}(y, f(\mathbf{x}))p(y, \mathbf{x})dyd\mathbf{x}$$

- However, we usually don't know $p(y, \mathbf{x})$, so we approximate this by the "empirical risk", defined over the training set

$$\mathcal{R}_{emp}[f] = \frac{1}{n}\sum_{i=1}^{n} \mathcal{L}(y, f(\mathbf{x}))$$

Minimising the empirical risk

- Consider a linear model that aims to predict the output (y) using a weighted combination of the inputs ($\mathbf{x}$)

$$f(\mathbf{x}, \mathbf{w}) = \mathbf{x}^T \mathbf{w} + b$$

- To estimate the weights we seek to minimise the empirical risk which is penalised to restrict model flexibility

$$\hat{\mathbf{w}} = \min_{\mathbf{w}} \sum_{i=1}^{n} \mathcal{L}(y_i, \mathbf{x}_i, \mathbf{w}) + \lambda J(\mathbf{w})$$

- Many algorithms (e.g. SVM, Lasso, ridge regression) are particular choices of $\mathcal{L}()$ and $J()$
- Probabilistic models can be viewed from a similar perspective

$$\log p(\mathbf{w}|\mathbf{y}, \mathbf{X}, \theta, \sigma) \propto \sum_{i=1}^{n} \log p(y_i|\mathbf{w}, \mathbf{x}_i, \sigma) + \log p(\mathbf{w}|\theta)$$

Probabilistic classification and regression

- The discriminant function approach is appealing and is often very efficient
- However, separating inference and decision also provides benefits, especially for classification

Advantages of probabilistic classification (Bishop, 2006)

- Minimizing risk (e.g. misclassification costs may change)
- Compensate for class priors (accommodate disease prevalence)
- "Reject option" (only make a decision if sufficiently confident)
- Combining classifiers
- Easily interpretable (predictive confidence)

Probabilistic prediction for clinical applications

Coherent handling of uncertainty is especially important in medicine

Sources of uncertainty in clinical applications

- Diagnostic uncertainty (class labels may be noisy)
- Heterogeneity in disease severity and course
- Individual variability in response to treatment

Probabilistic prediction for clinical applications

Coherent handling of uncertainty is especially important in medicine

Sources of uncertainty in clinical applications

- Diagnostic uncertainty (class labels may be noisy)
- Heterogeneity in disease severity and course
- Individual variability in response to treatment

In such applications predictive confidence is potentially highly informative about individual variability

$$p(y|\mathbf{x}) = 0.55\text{: ambiguous} \quad p(y|\mathbf{x}) = 0.99\text{: confident}$$

Outline

Introduction to Gaussian process models

GPs are flexible probabilistic kernel methods with many applications, e.g. classification and regression (Rasmussen and Williams, 2006a)

Advantages:

- Explicit probabilistic framework (Likelihood-Prior-Posterior)
- Natural extension to direct multi-class classification
- Provide mechanisms for automatic parameter optimisation (optimisation of Marginal Likelihood)

Gaussian process models

- With the GP framework, we can specify a wide range of **likelihoods** to measure data fit:

$$\text{Regression} : p(y_i|\mathbf{x}_i) = \mathcal{N}(f_i, \sigma^2) = f(\mathbf{x}_i, \mathbf{w}) + \sigma^2$$

$$\text{Binary Classification} : p(y_i = 1|\mathbf{x}_i) = \frac{1}{1 + \exp(-f_i)}$$

$$\text{Multi-Class Classification} : p(y_i = c|\mathbf{x}_i) = \frac{\exp(f_i^c)}{\sum_{c=1}^{C} \exp(f_i^c)}$$

Gaussian process models

- With the GP framework, we can specify a wide range of **likelihoods** to measure data fit:

$$\text{Regression} : p(y_i|\mathbf{x}_i) = \mathcal{N}(f_i, \sigma^2) = f(\mathbf{x}_i, \mathbf{w}) + \sigma^2$$

$$\text{Binary Classification} : p(y_i = 1|\mathbf{x}_i) = \frac{1}{1 + \exp(-f_i)}$$

$$\text{Multi-Class Classification} : p(y_i = c|\mathbf{x}_i) = \frac{\exp(f_i^c)}{\sum_{c=1}^{C} \exp(f_i^c)}$$

- GPs utilize a **Gaussian prior** to constrain the solution:

$$p(\mathbf{w}|\mathbf{X}, \theta) = \mathcal{N}(\mathbf{w}|\mathbf{0}, \mathbf{\Sigma}_p)$$

- We then compute the posterior distribution via Bayes rule

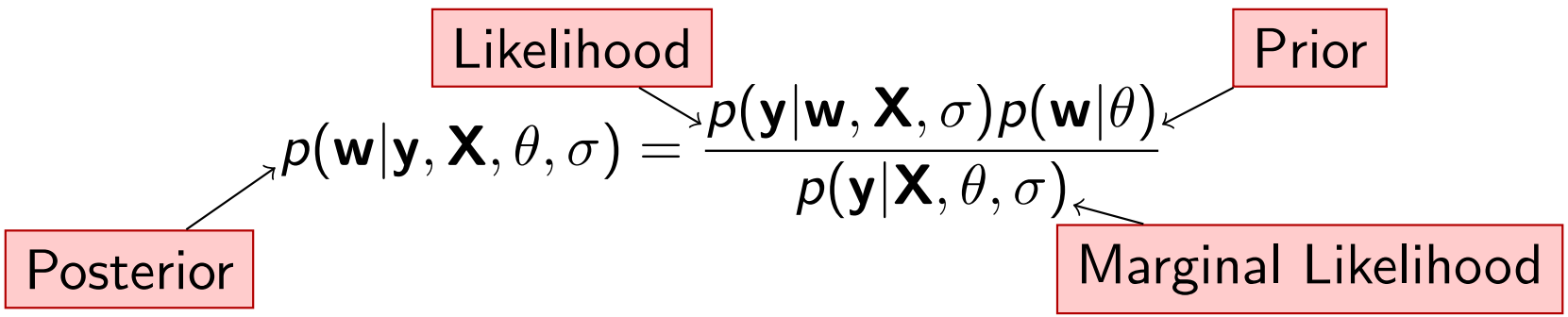
Weight space view

- There are two equivalent perspectives on GP models "weight" and "function" space

Weight space view

- There are two equivalent perspectives on GP models "weight" and "function" space

- Under the weight space view we are primarily interested in the posterior weight distribution:



Function space view

- Here we apply a Gaussian prior to the function values ($f_i = \mathbf{x}_i^T \mathbf{w}$) instead of the weights

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

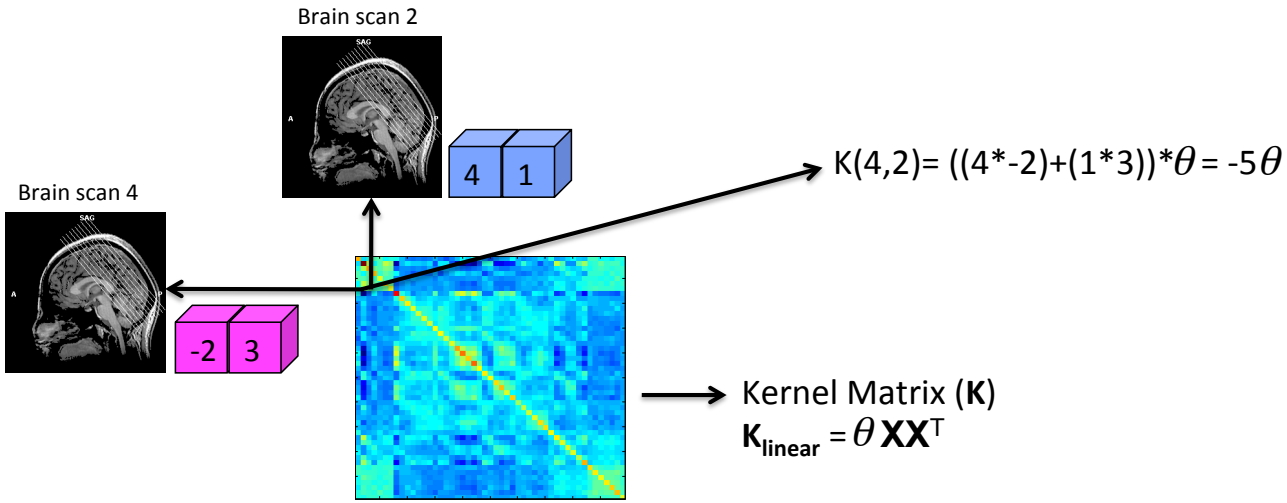
  where $\mathbf{K}$ is the covariance function of the prior.

- $\mathbf{K} = \mathbf{k}(\mathbf{x}_i, \mathbf{x}_j)$ is also referred to as the 'Kernel Function' and it encodes relationships between the function values over the input space

- We can use it to model linear and non-linear relationships.

Function space view

- **K** can be thought of in a similar way to the kernels in eg. SVM, ie. entry i, j is the similarity of two images



- In GPs, the value of the similarity for two images defines the prior knowledge of how similar the function values are
- As for other algorithms eg. Kernel Ridge Regression we tend to use a linear kernel in neuroimaging to avoid overfitting

Function space view: Regression

Say we want to predict a continuous measure such as age from our brain scans.

- Likelihood for homogenous Gaussian Noise:

$$P(y \mid f_i) = \mathcal{N}(f_i, \sigma^2)$$

Function space view: Regression

Say we want to predict a continuous measure such as age from our brain scans.

- Likelihood for homogenous Gaussian Noise:

$$P(y \mid f_i) = \mathcal{N}(f_i, \sigma^2)$$

- We perform inference on the function values using the likelihood and prior (Kernel Function) giving

$$\begin{aligned} f_{*\mu} &= \mathbf{k}_*^T (\mathbf{K} + \sigma^2 \mathbf{I})^{-1} \mathbf{y} \\ f_{*\sigma} &= k_{**} - \mathbf{k}_*^T (\mathbf{K} + \sigma^2 \mathbf{I})^{-1} \mathbf{k}_* \end{aligned}$$

- f_* is the function value at test point $\mathbf{x}_*$, $\mathbf{k}_*$ is the train-test kernel, k_{**} is the test-test kernel.

Function space view: Regression

Say we want to predict a continuous measure such as age from our brain scans.

- Likelihood for homogenous Gaussian Noise:

$$P(y \mid f_i) = \mathcal{N}(f_i, \sigma^2)$$

- We perform inference on the function values using the likelihood and prior (Kernel Function) giving

$$\begin{aligned} f_{*\mu} &= \mathbf{k}_*^T (\mathbf{K} + \sigma^2 \mathbf{I})^{-1} \mathbf{y} \\ f_{*\sigma} &= k_{**} - \mathbf{k}_*^T (\mathbf{K} + \sigma^2 \mathbf{I})^{-1} \mathbf{k}_* \end{aligned}$$

- f_* is the function value at test point $\mathbf{x}_*$, $\mathbf{k}_*$ is the train-test kernel, k_{**} is the test-test kernel.
- We take the prediction at test point $\mathbf{x}_*$ to be $y_{*\mu} = f_{*\mu}$ (as likelihood is Gaussian)
- Equivalent to Kernel Ridge Regression

Hyperparameter Estimation: Regression

- Log Marginal Likelihood has closed-form

$$\log P(\mathbf{y} \mid \mathbf{X}, \theta, \sigma) = -\frac{1}{2}\mathbf{y}^T(\mathbf{K}(\theta) + \sigma^2\mathbf{I})^{-1}\mathbf{y} - \frac{1}{2}\log\left|\mathbf{K}(\theta) + \sigma^2 I\right| - \frac{n}{2}\log 2\pi$$

- We maximise above to give hyperparameter estimates $\hat{\theta}, \hat{\sigma}$
- Plug them into the predictive equation

$$f_{*\mu} = \mathbf{k}_*^T(\mathbf{K}(\hat{\theta}) + \hat{\sigma}^2\mathbf{I})^{-1}\mathbf{y}$$

- The optimisation of marginal likelihood distinguishes GP regression from Kernel Ridge Regression in practice

Multi-Class Classification using GPs

Say we want to predict clinical groups eg. Controls/Unipolar Depression/Schizophrenia from our brain scans.

- For multi-class classification into C possible classes $y = 1, \ldots, C$ we use the following likelihood:

Weight-Space

$$p(y_i = c \mid \mathbf{x}_i, \mathbf{w}) = \frac{\exp(\mathbf{x}_i^T \mathbf{w^c})}{\sum_{c=1}^{C} \exp(\mathbf{x}_i^T \mathbf{w^c})}$$

Function-Space

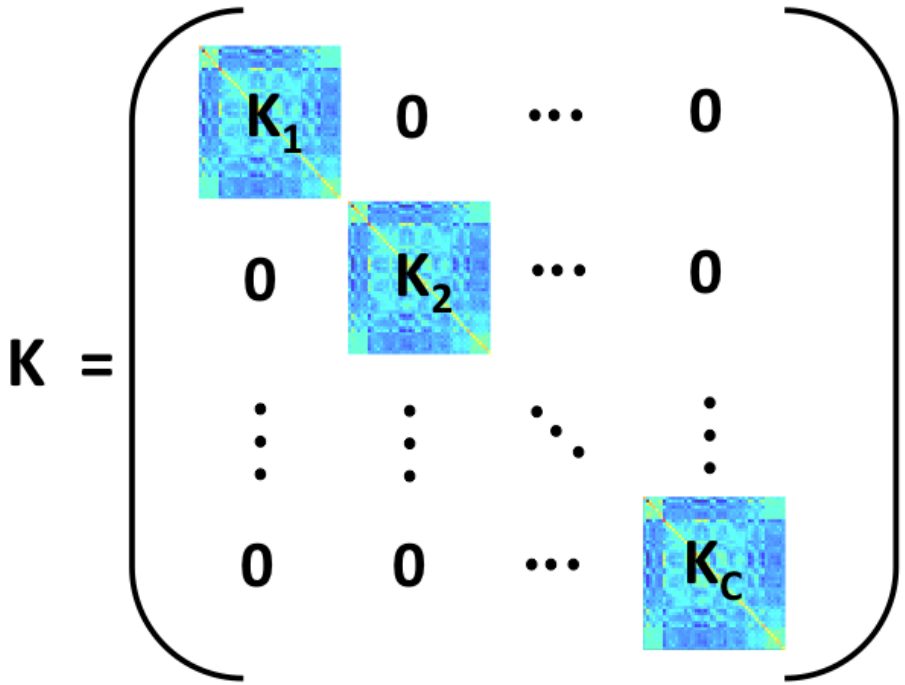
$$p(y_i = c \mid \mathbf{f_i}) = \frac{\exp(f^c)}{\sum_{c=1}^{C} \exp(f^c)}$$

- Weight vector parameter $\mathbf{w}$ consists of C weight vectors (1 per class), and similarly for function values $\mathbf{f}_i$:

$$\begin{aligned} \mathbf{w} &= [\mathbf{w}^1, \mathbf{w}^2, \ldots, \mathbf{w}^C] \\ \mathbf{f}_i &= [\mathbf{x}_i^T \mathbf{w}^1, \mathbf{x}_i^T \mathbf{w}^2, \ldots, \mathbf{x}_i^T \mathbf{w}^C] \\ &= [f_i^1, f_i^2, \ldots, f_i^C] \end{aligned}$$

Multi-Class Classification using GPs

- The kernel function (prior) for the function values is now a block-diagonal matrix $\mathbf{K} = [\mathbf{K}_1 \mathbf{K}_2 \dots \mathbf{K}_C]$



- In general the kernels $\mathbf{K}_c$ for each class do not need to be equal
- In PRoNTo we use linear kernels for each $\mathbf{K}_c$

Inference for Multi-Class Classification

- Unlike GP Regression, inference requires approximation techniques
- The 'Laplace' approximation gives

$$\mathbf{f}_{*\mu} = \mathbf{Q}_*^T(\mathbf{y} - \hat{\boldsymbol{\pi}})$$
$$\mathbf{f}_{*\Sigma} = \text{diag}(\mathbf{k}(\mathbf{x}_*, \mathbf{x}_*)) - \mathbf{Q}_*^T(\mathbf{K} + \mathbf{W}^{-1})^{-1}\mathbf{Q}_*$$

where

$$\mathbf{Q} = \begin{pmatrix} \mathbf{k}_1(\mathbf{x}_*) & \mathbf{0} & \dots & \mathbf{0} \\ \mathbf{0} & \mathbf{k}_2(\mathbf{x}_*) & \dots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \dots & \mathbf{k}_C(\mathbf{x}_*) \end{pmatrix}$$

- Here, $\mathbf{W}, \hat{\boldsymbol{\pi}}$ are parameters associated/derived with Laplacian inference

Predictions for Multi-Class Classification

- We now have the distribution of function values $\mathbf{f}_* = [f_*^1, f_*^2, \ldots, f_*^C]$ for each possible class at a test point $\mathbf{x}_*$
- A class probability vector $\bar{\boldsymbol{\pi}}$ for the testpoint can be given by sampling: (Below taken from Rasmussen and Williams (2006a))

$\boldsymbol{\pi}_* := \mathbf{0}$ initialize Monte Carlo loop to estimate
1. **for** $i := 1 : S$ **do** predictive class probabilities using S samples
   $\mathbf{f}_* \sim \mathcal{N}(\mathbf{f}_{*\mu}, \mathbf{f}_{*\Sigma})$ sample latent values from joint Gaussian posterior
2. $\boldsymbol{\pi}_* := \boldsymbol{\pi}_* + \exp(f_*^c) / \sum_{c'} \exp(f_*^{c'})$ accumulate probability eq. (3.34)

   end for
3. $\bar{\boldsymbol{\pi}}_* := \boldsymbol{\pi}_* / S$ normalize MC estimate of prediction vector

- Results in a vector of class probabilities $\bar{\boldsymbol{\pi}}_*$

Predictions for Multi-Class Classification

- For a given test point $\mathbf{x}_*$, we now have a vector of probabilities for each class e.g.

$$\bar{\boldsymbol{\pi}}_* = [0.8, 0.05, 0.15]$$

  In the above case, we might choose a 'hard' assignment to class 1 eg. test subject is a 'Control'.

- We could have a situation like below:

$$\bar{\boldsymbol{\pi}}_* = [0.31, 0.34, 0.35]$$

  A hard assignment would choose class 3 eg. 'Schizophrenia', but it is not as convincing as the first case. We could 'reject' a hard assignment here and say we are undecided due to the large degree of uncertainty.

Hyperparameter Estimation for Multi-Class Classification

- We use the Laplace approximation for the Marginal Likelihood

$$\log P(\mathbf{y} \mid \mathbf{X}, \theta) = -\frac{1}{2}\hat{\mathbf{f}}\mathbf{K}^{-1}\hat{\mathbf{f}} + \mathbf{y}^T\hat{\mathbf{f}} - \sum_{i=1}^{n}\log(\sum_{c=1}^{C}\exp \hat{f}_i^c) - \frac{1}{2}\log\left|I_{Cn} + W^{\frac{1}{2}}KW\frac{1}{2}\right|$$

- We optimise the above expression to determine kernel parameters θ and plug them into predictive equations.

Relevance Vector Machines

- The relevance vector machine is a type of sparse Bayesian model for regression and classification (Tipping, 2001)
- For regression, the RVM uses the same Gaussian likelihood as the GP and applies a prior over the weights of the form:

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

- The α_i are scaling parameters which determine the "relevance" of each sample or voxel (MacKay, 2003). These are given flat Gamma priors.

Relevance Vector Machines

- The relevance vector machine is a type of sparse Bayesian model for regression and classification (Tipping, 2001)
- For regression, the RVM uses the same Gaussian likelihood as the GP and applies a prior over the weights of the form:

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

- The α_i are scaling parameters which determine the "relevance" of each sample or voxel (MacKay, 2003). These are given flat Gamma priors.
- The RVM forces the posterior probability for the weights to concentrate on only a few of the samples/voxels. Samples/voxels with a low weight are pruned from the model ($\rightarrow$ **Sparsity**)
- The RVM is not solvable in closed form and requires numerical approximation(s) to the posterior distribution

Outline

Conclusions

- Probabilistic approaches to pattern classification are complementary to alternative methods
- They share many features with conventional approaches (e.g. penalised linear models)
- They aim to be honest about uncertainty at all stages of analysis (**coherence**)
- This provides a number of advantages, especially for clinical applications, e.g.:
  - Provide a natural way to include existing information (priors)
  - To compensate for variable class frequencies
  - To represent variabilities in illness severity
- However they also have disadvantages
  - Estimating probability distributions requires more computation than just estimating a decision function.
  - Some methods may not scale as well to large datasets ($O(n^3)$)

References

Christopher Bishop. *Pattern Recognition and Machine Learning*. Springer, New York, 2006.

D MacKay. *Information Theory, Inference and Learning Algorithms*. Cambridge University Press, Cambridge, U.K., 2003.

C. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning*. The MIT Press, Cambridge, Massachusetts, 2006a.

Carl E. Rasmussen and Christopher Williams. *Gaussian Processes for Machine Learning*. MIT Press, 2006b. URL `http://www.gaussianprocess.org/gpml/`.

P M Rasmussen, L K Hansen, K H Madsen, N W Churchill, and S C Strother. Model sparsity and brain pattern interpretation of classification models in neuroimaging. *Pattern Recognition*, 45:2085–2100, 2011.

M Tipping. Sparse bayesian learning and the relevance vector machine. *Journal of Machine Learning Research*, 1: 211–244, 2001.