Learning from Data: Regression

Amos Storkey, School of Informatics

November 3, 2005

http://www.anc.ed.ac.uk/~amos/lfd/
Classification or Regression?

- Classification: want to learn a discrete target variable.
- Regression: want to learn a continuous target variable.
- Linear regression, generalised linear models, and nonlinear regression.
- Most regression models can be turned into classification models using the logistic trick of logistic regression.
One Dimensional Data
Simple example: 1 dimensional linear regression.

Suppose we have data of the form \((x, y)\), and we believe the data should follow a straight line.

However we also believe the target values \(y\) are subject to measurement error, which we will assume to be Gaussian.

Often use the term *error measure* for the negative log likelihood.

Hence training error, test error.

Remember: Gaussian noise results in a quadratic negative log likelihood.
Believe the data should have a straight line fit: $y = a + bx$

but that there is some measurement error for $y$: $y = a + bx + \eta$ where $\eta$ is a Gaussian noise term.

Training error is

$$- \sum_{\mu} \log P(\eta = (y^\mu - bx^\mu - a)) = A \sum_{\mu} (y^\mu - bx^\mu - a)^2 + B.$$ 

for training data $\{(x^\mu, y^\mu); \mu = 1, \ldots, N\}$ of size $N$. $A$ and $B$ depend on the variance of the Gaussian, but do not actually matter in a minimisation problem: we get the same minimum whatever $A$ and $B$ are.
Linear Regression

Generated Data

Amos Storkey, School of Informatics
Learning from Data: Regression
Consider the case where we are interested in \( y = f(x) \) for \( D \) dimensional \( x \): \( y = a + b^T x \)

In fact if we set \( w = (a, b^T)^T \) and introduce \( \phi = (1, x^T)^T \), then we can write

\[
y = w^T \phi
\]

for the new augmented variables.

The training error (up to an additive and multiplicative constant) is then

\[
E(w) = \sum_{\mu=1}^{N} (y^\mu - w^T \phi^\mu)^2
\]

where \( \phi^\mu = (1, (x^\mu)^T)^T \).
Minimum training error equals maximum log-likelihood.

Take derivatives of the training error:

\[ \nabla_w E(w) = 2 \sum_{\mu=1}^{N} \phi^\mu (w^T \phi^\mu - y^\mu) \]

Write \( \Phi = (\phi^1, \phi^2, \ldots, \phi^N) \), and \( y = (y^1, y^2, \ldots, y^N)^T \).

Then

\[ \nabla_w E(w) = 2\Phi (\Phi^T w - y) \]
Setting the derivatives to zero to find the minimum gives

$$\Phi \Phi^T w = \Phi y$$

This means the maximum likelihood $w$ is given by

$$w = (\Phi \Phi^T)^{-1} \Phi y$$

The term $(\Phi \Phi^T)^{-1} \Phi$ is called the *pseudo-inverse*. 
The black line is the maximum likelihood fit to the data.
Recap

- Error measure is the negative log likelihood
- Gaussian error term is the sum-squared error (up to a multiplicative and additive constant).
- Write down the regression error term.
- Build weight vector $\mathbf{w}$ and data vector $\phi$.
- Take derivatives and set to zero to obtain pseudo-inverse solution.
But...

▶ All this just used $\phi$.
▶ We chose to put the $\mathbf{x}$ values in $\phi$, but we could have put anything in there, including nonlinear transformations of the $\mathbf{x}$ values.
▶ In fact we can choose any useful form for $\phi$ so long as the final derivatives are linear in $\mathbf{w}$. We can even change the size.
▶ We already have the maximum likelihood solution in the case of Gaussian noise: the pseudo-inverse solution.
▶ Models of this form are called generalized linear models or linear parameter models.
Example: polynomial fitting

- Model $y = w_1 + w_2x + w_3x^2 + w_4x^3$.  
- Set $\phi = (1, x, x^2, x^3)^T$ and $w = (w_1, w_2, w_3, w_4)$.  
- Can immediately write down the ML solution: $w = (\Phi\Phi^T)^{-1}\Phi y$, where $\Phi$ and $y$ are defined as before.
Higher dimensional outputs

- Suppose the target values are vectors $y$.
- Then we introduce different $w_i$ for each $y_i$.
- Then we can do regression independently in each of those cases.
Set $\phi_i(x) = \exp\left(-\frac{1}{2}(x - m_i)^2/\alpha^2\right)$.

Need to position these “basis functions” at some prior chosen centres $m_i$ and with a given width $\alpha$. We will discuss how the centres and widths can also be considered as parameters in a future lecture.

Finding the weights is the same as ever: the pseudo-inverse solution.
How many radial basis bumps do we need?
Suppose we only needed 3 for a 1D regression problem.
The we would need $3^D$ for a D dimensional problem.
This becomes large very fast: this is commonly called the curse of dimensionality.
How do we compare different models?

For example we could introduce 1, 2, ..., 4000 radial basis functions.

The more parameters the model has, the better it will do.

Models with huge numbers of parameters could fit the training data perfectly.

Is this a problem?
Summary

- Lots of different models are linear in the parameters.
- For regression models, the maximum likelihood solution is analytically calculable.
- The optimum value is given by the pseudo-inverse solution.
- Overfitting.