

COMP4702/COMP7703 - Machine Learning

Prac 3 – Regression and Parametric models in Matlab

Aims:

- To gain some experience in performing regression with linear and polynomial models and classification with parametric models.
- To gain familiarity with Matlab.
- To produce some assessable work for this subject.

Procedure:

Introduction to Matlab

Matlab is a general-purpose commercial scientific programming environment. Some of you will doubtless be familiar with matlab, but if not, it shouldn't take you very long to learn enough to do this prac. On the course material webpage you will find a "Matlab primer" document:

An Introduction to Matlab for Cognitive Programming

which should be a helpful reference.

- Open Matlab and refer to the example presented in fig.4.5 of the text (slide 23 in lecture notes). To start, we will recreate this example.
 - Plot the function ($f(x) = 2\sin(1.5x)$) in the range shown.
 - Create a "sample training set" of 20 points as indicated in the text. That is, generate a random set of x-values, and corresponding outputs by adding Gaussian random noise (using `randn()`) to the function above. Plot the data over your sine wave plot (use the "hold on" command).
- Now we will fit some models to this data. To implement linear and polynomial regression, matlab has a curve fitting toolbox and curve fitting tool – to run it type "cftool". Refer to the matlab help to learn how to use it.
- Fit polynomials of increasing order to the data and observe the models fitting the data and the output in the results box on the cftool window.
- **Q1:** Using the curve fitting tool to produce a plot of error as a function of model order up to 9, similar to that shown in fig.4.7 (slide 25 in notes). To do this you will need to create a second (*validation*) dataset (in addition to the *training* set above). Some hints:
 - The cftool results show you the SSE (sum of squares error = $N \cdot \text{MSE}$ where N is the number of data points). It also shows the trained model (polynomial with coefficients). You can paste this model back into the matlab command line (or a script or function) to calculate validation set error.

- Observe the effect of overfitting as the model complexity increased while you trained the models. To see this clearly, you might need to increase the variance of the additive noise to the function when generating the outputs of your training data (the book uses variance of 0.1).

Temperature Dataset

On the course material webpage you will find a dataset “globtp.dat”. This dataset comes from:

<http://www.robhyndman.info/TSDL/meteorology.html>

and records annual changes in global “surface air” temperature from 1880 – 1995.

- **Q2:** Fit polynomials of varying degree to this data. Do you see any overfitting? Do higher-order polynomials achieve lower error?

Parametric Probabilistic Classification

Using matlab, write a function or script that implements a simple parametric classifier to produce plots similar to Figs 4.2 and 4.3 (slide 13 and 14 of the lecture notes for Chap. 4). More specifically:

- Your code should take as input data for a one-dimensional, two-class classification problem, assuming a real-valued input feature.
- You should use Gaussian (normal) distributions as your model; i.e. fit a Gaussian to each class distribution using maximum likelihood estimates for the parameters of each Gaussian. This produces a model for $p(x|C1)$ and for $p(x|C2)$.
- Assume equal priors for the class distributions ($P(C1)$ and $P(C2)$). In this case, the posteriors are just $p(C1|x)$ and $p(C2|x)$ and are equal to the discriminants $g(x)$ for each class.
- To test your code, create a “reduced” version of the Iris dataset (see Prac 1 – a plain text version of this dataset can be found on the course material webpage); firstly by discarding all the features from the data except the first one (sepal length), secondly by discarding all examples for the third class (Iris-virginica).
- Using the matlab `hist()` function, produce histograms to give you a picture of the data and something to compare you results with.
- **Q3:** Use your code to produce plots of the likelihoods and class posteriors for the reduced iris data.