Machine learning Lab 5:

Iustina Ivanova, *Msc student Artificial Intelligence,* email: *ii1n17@soton.ac.uk*

***Abstract*—This Lab is consisted of 2 parts: 1) radial basis functions and their perfomance; 2) the comparison between RBF and linear regression. For calculation the Boston Houses dataset was used.**

Keywords—RBF, neural network, linear regression, Boston houses regression.

I. Introduction

This document is an introduction to linear neural networks, particularly radial basis function (RBF) networks. Linear models have been studied in statistics for about 200 years and the theory is applicable to RBF networks which are just one particular type of linear model.

A. Construct a radial basis functions

A linear model for a function y(x) takes form

$f = \overrightarrow{w}\overrightarrow{x} + w_0$

The model f is expressed as a linear combination of a set of m fixed functions (often called *basis functions* by analogy with the concept of a vector being composed of a linear combination of basis vectors). Radial basis functions are a special class of function. Their characteristic feature is that their response decreases (or increases) monotonically with distance from a central point. The centre, the distance scale, and the precise shape of the radial function are parameters of the model, all fixed if it is linear.

A typical radial function is the Gaussian which, in the case of a scalar input, is

$h(x) = exp(-\frac{(x-c)^2}{r^2})$.

Its parameters are its centre c and its radius r. Gaussian-like RBFs are local (give a significant response only in a neighbourhood near the centre) and are more commonly used.

Radial functions are simply a class of function. In principle, they could be employed in any sort of model (linear or non-linear) and any sort of network (single-layer or multi-layer). However, radial basis function networks have traditionally been associated with radial functions in a single-layer network such as shown in Figure 1.

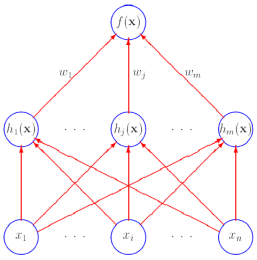

Figure 1. The traditional radial basis function network. Each of n components of the input vector x feeds forward to m basis functions whose outputs are linearly combined with weights $\{w_j\}_{j=1}^m$ into the network output $f(x)$.

When applied to supervised learning with linear models the least squares principle leads to a particularly easy optimisation problem. If the training set is $\{(x_i, \widehat{y}_i)\}_{i=1}^p$, then the least squares recipe is to minimise the sum-squared-error

$S = \sum_{i=1}^p \left(\widehat{y}_i - f\left(x_i\right)\right)^2$

with respect to the weights of the model. If a weight penalty term is added to the sum-squared-error, then the following *cost function* is minimised

$C = \sum_{i=1}^p \left(\widehat{y}_i - f\left(x_i\right)^2\right) + \sum_{j=1}^m \lambda_j w_j^2,$

where the $\{\lambda_j\}_{j=1}^m$ are regularisation parameters.

The RBF model in this Lab is given by

$g(x) = \sum_{k=1}^K \lambda_k \phi\left(\|x - c_k\|\right)$

This is a model in which the nonlinear part is fixed and only the weights λ_k are estimated in a manner similar to linear regression. The nonlinear part is fixed in some sensible way (K-means clustering was used to do this).

B. The optimal Weight Vector

The minimisation of the cost function leads to a set of m simultaneous linear equations in the m unknown weights and how the linear equations can be written more conveniently as the matrix equation

$\mathbf{A}\widehat{w} = \mathbf{H}^T\widehat{y}$,

where $\mathbf{H}$, the *design matrix*, is

$$\mathrm{H} = \begin{bmatrix} h_1(\mathbf{x}_1) & h_2(\mathbf{x}_1) & \dots & h_m(\mathbf{x}_1) \\ h_1(\mathbf{x}_2) & h_2(\mathbf{x}_2) & \dots & h_m(\mathbf{x}_2) \\ \vdots & \vdots & \ddots & \vdots \\ h_1(\mathbf{x}_p) & h_2(\mathbf{x}_p) & \dots & h_m(\mathbf{x}_p) \end{bmatrix}, \quad (1)$$

$\mathbf{A}^{-1}$, the *variance matrix*, is

$\mathbf{A}^{-1} = \left(\mathbf{H}^T\mathbf{H} + \mathbf{\Lambda}\right)^{-1}$,

the elements of the matrix $\mathbf{\Lambda}$ are all zero except for the regularisation parameters along its diagonal and $\widehat{y} = [\widehat{y}_1, \widehat{y}_2, ...\widehat{y}_p]^T$ is the vectore of training set outputs. The solution is the so-calles $normalequation$,

$\widehat{\mathbf{w}} = \mathbf{A}^{-1}\mathbf{H}^T\widehat{\mathbf{y}}$,

and $\widehat{\mathbf{w}} = [\widehat{w}_1, \widehat{w}_2...\widehat{w}_3]^T$ is the vector of weights which minimises the cost function.

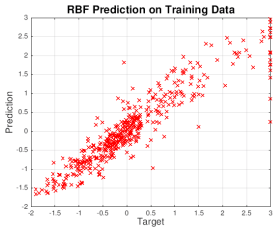

Fig. 2: RBF prediction on Training Data.

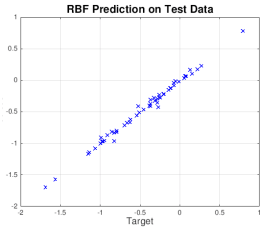

Fig. 3: RBF prediction on Test Data.

C. Boston Houses Dataset RBF perfomance

The Boston Houses Dataset was splited to training set (80%) and test set (20%). For this data model predictions is shown in figure 2.

The computed model was used for prediction of test data: figure 3 shows that test data is similar to training data. There are two stage in training procedure:

1) unsupervised training to determine the parameters of the basis function

2) by fixing the parameters of the basis function algorithm determine the weights (w_{kj}) by Supervised training.

If we consider a clustering algorithm for which the number of clusters is predefined (K-Means), then algorithm is :

1) set the number of cluster to M and run the clustering algorithm

2) set the centers of the basis functions equals to the centers of clusters

3) set the widths (variances) of the basis functions eqauls to the variances of clusters.

D. The model complexity

Choosing a very simple model may give rise to poor results (M=1) and choosing a very complex model may give rise to over-fitting and thus poor generalization performance.

The differences between errors of training and test data at different values of the number of basis functions, K, is shown in figure 4.

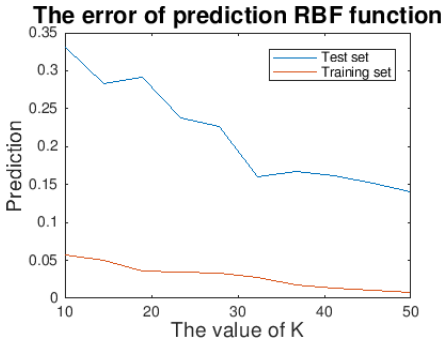

Figure 4. The performance of errors of different K.

The higher amount of clusters give less mistakes.

E. Performance of RBF and linear regression.

The comparison with linear regression shows that RBF give a better solution.

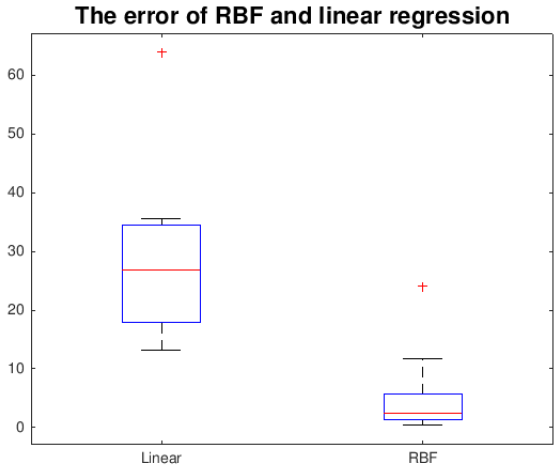

Figure 5. The results of RBF and linear regression.

The boxplot diagram was calculated for K different K values for RBF (from 1 to 15). The mean squared error on test data is obtained from 10 random partition- ings. The average error that RBF gives is smaller than linear regression.

REFERENCES

[1] R. O. Duda, P. E. Hart, D. G. Stork *Pattern Classification*, 2nd ed.

[2] M. J. L. Orr *Introduction to Radial Basis Function Networks*, April 1996.