Homework#6

Submitted by:Veera Venakata Sathya Bhargav Nunna

The radial basis function neural network architecture has been implemented with two input neurons, n hidden rbf neurons and one neuron in the output layer with two methods of activation function.

Several experiments have been run with different number of neurons in the hidden layer. Below are the plots for the hidden layer with 4 and 11 neurons.

Method 1:

The radial basis function neuron activation function is

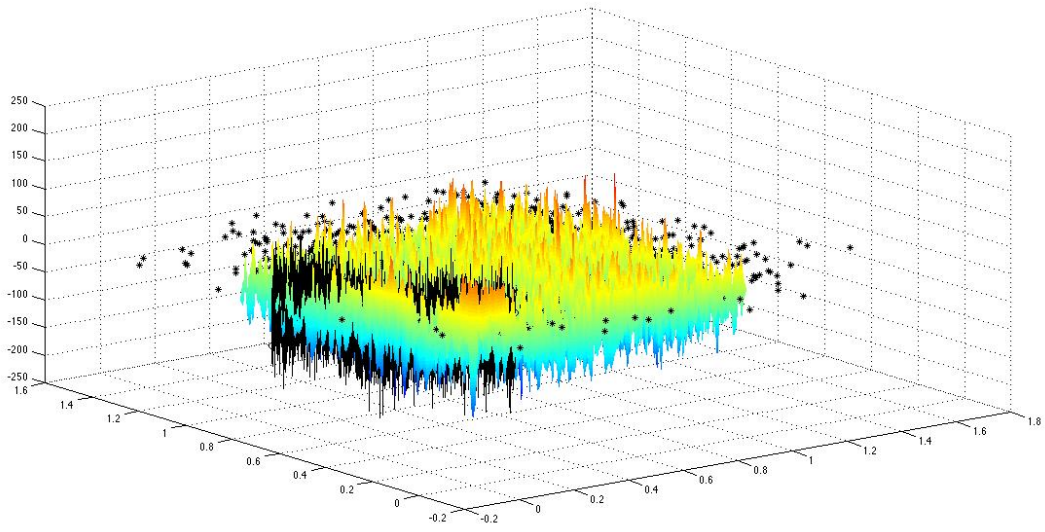
exp(|x-t|2/2*sq of sd)

x is the input vetor, t is the centeroid of the cluster, sd is  the standard deveation.

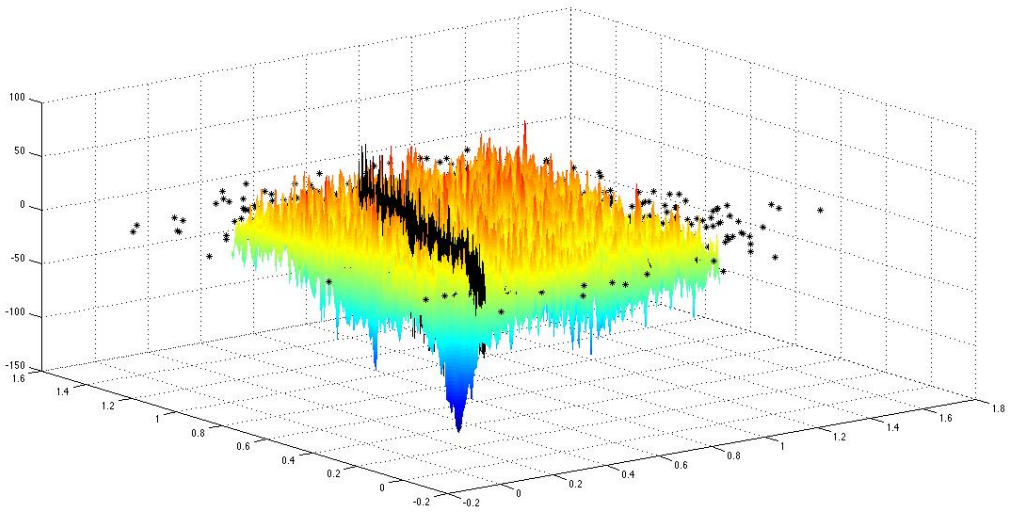
Performing kmeans clustering for the training data

Generalization plot:

The surface plots for k=4



The surface plot for k=11



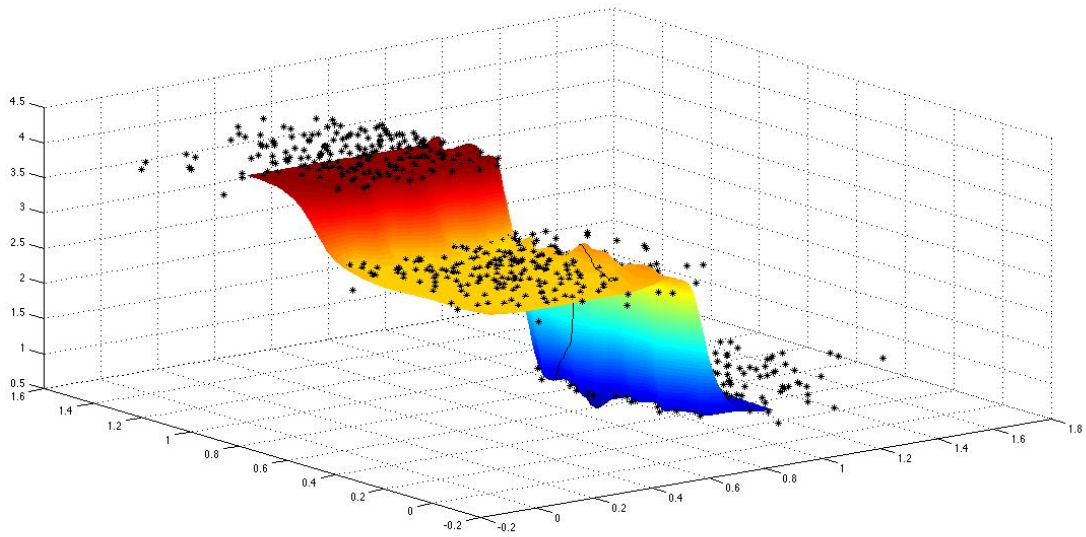

Method 2:

The activartion function for the radial basis function neuron is

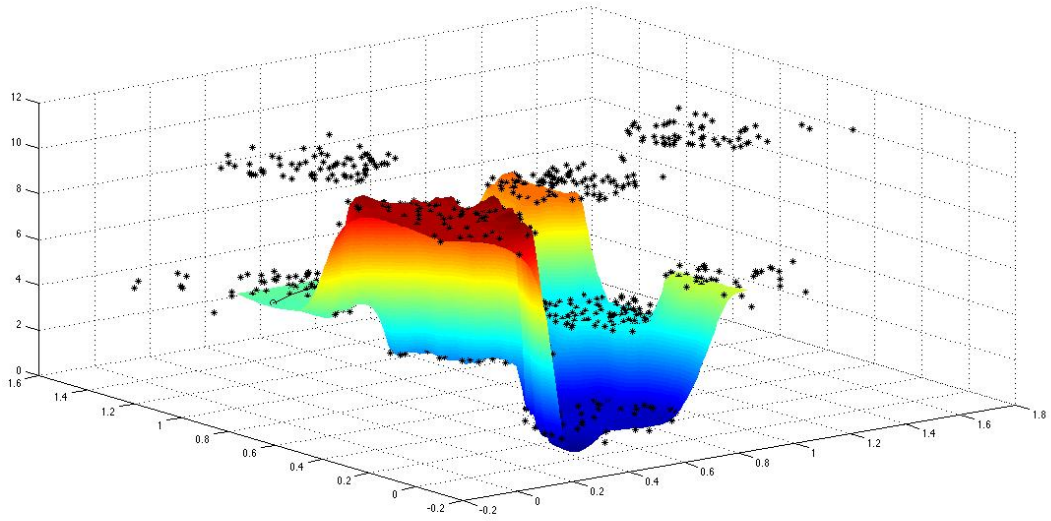
Exp(-1\2(|X-t|)Sigma(|X-t|)

Generalization plot:

Peroforming kmeans clustering for k=4




Performing kmeans clustering for k=11



Appendix:

```
#include<iostream.h>

#include<math.h>

#include<fstream.h>


using namespace std;


int main()

{

 int i;

cout<<"Enter number of neuron\n";

cin>>i;


double inputx1[800]={ //Input data x1};

double inputx2[800]={//Input data for x2 };

double d[800]={//desired values};


FILE *fp;

if((fp = fopen("nntrain_cluster.csv", "r+")) == NULL) {

cout<<"No file! \n";

exit(1);

}

for(int j=0; j<800; j++) {

fscanf(fp, "%f,%f,%f\n",&inputx1[j], &inputx2[j], &d[j]);

cout<<"Read success.."<<j<<"\n";

}
```

```
double sigma[7]={0.1757,0.1571,0.1512,0.1311,0.1592,0.1522,0.1891};
//double sigma[i];
double tx[7]={0.2721,0.6428,1.1621,0.6238 ,1.0835 ,0.9242 ,0.2964};
double ty[7]={0.9893,1.0674,0.5696,0.5461,1.0743,0.2684,0.3233};
//double tx[7]={};
//double ty[7]={};
double rbfout[i];
double oi1[1];
double oi2[1];
double wo[i];
double n=0.1;
for(int k=0;k<i;k++)
{
	wo[k]=1;
}
cout<<"Data sets initialized..";

double sum=0;
double ersum=0;
for(int j=0;j<800;j++)
{

	oi1[0]=inputx1[j];
	oi2[0]=inputx2[j];
	for(int k=0;k<i;k++) //RBF neurons activations
```

```
    {
      double ed=((ty[k]-oi2[0])*(ty[k]-oi2[0]))+((tx[k]-oi1[0])*(tx[k]-oi1[0]));

      rbfout[k]=exp(ed)/(2*(sigma[k]*sigma[k]));

//      cout<<"rbf act\n";
    }

    for(int k=0;k<i;k++) //activation for output neuron
    {
       sum=(rbfout[k]*wo[k])+sum;
      // cout<<"Sig sq:  "<<sum<<"\n";
    }

    sum=1/1+exp(sum);
    cout<<j<<",";
    cout<<sum<<",";
    cout<<d[j]<<"\n";
    for(int k=0;k<i;k++)  //error correction for output neuron
    {
    ersum=n*(d[j]-sum)*rbfout[k];
    //cout<<"Error: "<<ersum<<"\n";
    wo[k]=wo[k]+ersum;
    }
    sum=0;
    ersum=0;
```

```
}
for(int k=0;k<i;k++)
 {
       //cout<<"Weights: "<<wo[k]<<"\n";
 }


 return 0;

}
```