MIDTERM ECE750/SDE625

Yanxin Wang

Solution of Problem 1

a) From n-space to n-1 space there are n projections. From n-space to n-2 space, there are $C_n^{n-2}=\frac{n(n-1)}{2!}$ projections. Similarly, from n-space to n-r space, there are $C_n^{n-r}=\frac{n(n-1)\cdots(n-r+1)}{r!}$ projections. Hence, the total number of different projections is:

$$P_n=n+\frac{n(n-1)}{2!}+\frac{n(n-1)(n-2)}{3!}+\cdots+\frac{n(n-1)(n-2)\cdots\times 3\times 2\times 1}{r!}$$

Now consider the given three-dimensional fuzzy relation $R(x_i,y_j,z_k)$.

Projection on X: $R_1=\sum_i \frac{\mu_{R_1}(x_i)}{x_i}=\frac{0.6}{x_1}+\frac{1.0}{x_2}+\frac{0.8}{x_3}=\begin{bmatrix}0.6\\1.0\\0.8\end{bmatrix}$

Projection on Y: $R_2=\sum_i \frac{\mu_{R_2}(y_i)}{y_i}=\frac{0.6}{y_1}+\frac{1.0}{y_2}+\frac{0.8}{y_3}=\begin{bmatrix}0.6\\1.0\\0.8\end{bmatrix}$

Projection on Z: $R_3=\sum_i \frac{\mu_{R_3}(z_i)}{z_i}=\frac{0.8}{z_1}+\frac{1.0}{z_2}+\frac{0.7}{z_3}=\begin{bmatrix}0.8\\1.0\\0.7\end{bmatrix}$

Projection on $X\times Y$: $R_4(x_i,y_j)=\underset{z_k}{Max}\,R(x_i,y_j,z_k)=\begin{bmatrix}0.6&0.5&0.3\\0.4&1.0&0.6\\0.2&0.6&0.8\end{bmatrix}$

Projection on $Y\times Z$: $R_5(y_j,z_k)=\underset{x_i}{Max}\,R(x_i,y_j,z_k)=\begin{bmatrix}0.5&0.6&0.4\\0.8&1.0&0.7\\0.6&0.8&0.5\end{bmatrix}$

Projection on $Z\times X$: $R_6(z_k,x_i)=\underset{y_j}{Max}\,R(x_i,y_j,z_k)=\begin{bmatrix}0.5&0.8&0.6\\0.6&1.0&0.8\\0.4&0.7&0.5\end{bmatrix}$

There are six different projections in total.

b)

(1)Non-decreasing Property:

For the given function, we have

$$x_1 T y_1 = \max[0, x_1 + y_1 - 1]$$

$$x_2 T y_2 = \max[0, x_2 + y_2 - 1]$$

If $0 \le x_1 \le x_2 \le 1$ and $0 \le y_1 \le y_2 \le 1$, then $x_1 + y_1 - 1 \le x_2 + y_2 - 1$

Hence, RHS of the first equation $\le$ RHS of the second equation above.

Hence, $x_1 T y_1 \le x_2 T y_2$

(2)Commutative Property

For the given function, we have

$$xTy = \max[0, x + y - 1]$$

$$yTx = \max[0, y + x - 1]$$

The RHS of the two equations are identical. Hence $xTy = yTx$.

(3)Associative Property

For the given function, we have

$$(xTy)Tz = \max[0, \max\{0, x + y - 1\} + z - 1]$$

Use the fact that

$$\max[a, b] + c = \max[a + c, b + c]$$

$$\max[a, \max\{b, c\}] = \max[a, b, c]$$

Hence

$$\begin{aligned}(xTy)Tz &= \max[0, \max\{z - 1, x + y + z - 2\}] \\ &= \max[0, z - 1, x + y + z - 2] \\ &= \max[0, x + y + z - 2]\end{aligned}$$

(because $z \le 1$)

Similarly,

$$\begin{aligned}xT(yTz) &= \max[0, \max\{0, y + z - 1\} + x - 1] \\ &= \max[0, \max\{x - 1, x + y + z - 2\}] \\ &= \max[0, x - 1, x + y + z - 2] \\ &= \max[0, x + y + z - 2]\end{aligned}$$

(because $x \le 1$)

Hence

$(xTy)Tz = xT(yTz)$

(4)Boundary Conditions

For the given function, we have

$$\begin{aligned}xT1 &= \max[0, x + 1 - 1] \\ &= \max[0, x] \\ &= x\end{aligned}$$

Also, we have

$$xT0 = \max[0, x+0-1]$$
$$= \max[0, x-1]$$
$$= 0$$

(because $x \le 1$)

So $\max[0, x+y-1]$ is a t-norm.

Its corresponding t-conorm could be determined by De Morgan's Law

$$xSy = 1-(1-x)T(1-y)$$
$$= 1-\max[0, 1-x+1-y-1]$$
$$= -\max[-1, 1-x-y-1]$$
$$= -\max[-1, -x-y]$$
$$= \min[1, x+y]$$

So $\min[1, x+y]$ is its corresponding t-conorm.

Solution of Problem 2

a) For the given conditions, we have

$$w_{k+1} = w_k - \eta \left.\frac{\partial E(w)}{\partial w}\right|_{(k)}$$
$$= w_k - \eta \left.\frac{\partial 0.5 w^T Q w}{\partial w}\right|_{(k)}$$
$$= w_k - 0.5\eta (Qw + Q^T w)\Big|_{(k)}$$
$$= w_k - \eta Q w_k$$

Hence,

$$E(w_{k+1}) = 0.5 w_{k+1}^T Q w_{k+1}$$
$$= 0.5\left(w_k - \eta Q w_k\right)^T Q \left(w_k - \eta Q w_k\right)$$
$$= 0.5\left(w_k^T - \eta w_k^T Q^T\right) Q \left(w_k - \eta Q w_k\right)$$
$$= 0.5\left(w_k^T Q - \eta w_k^T Q^T Q\right)\left(w_k - \eta Q w_k\right)$$
$$= 0.5\left(w_k^T Q - \eta w_k^T Q^2\right)\left(w_k - \eta Q w_k\right)$$
$$= 0.5\left(w_k^T Q w_k - \eta w_k^T Q^2 w_k - \eta w_k^T Q^2 w_k + \eta^2 w_k^T Q^3 w_k\right)$$
$$= 0.5(w_k^T Q w_k - 2\eta w_k^T Q^2 w_k + \eta^2 w_k^T Q^3 w_k)$$

If η is the optimal learning rate at k, then we have $E(w_{k+1}) \to 0$.

Hence,

$$w_k^T Q w_k - 2\eta w_k^T Q^2 w_k + \eta^2 w_k^T Q^3 w_k = 0$$

It is similar to a linear algebraic equation. We can solve it in the same way.

For a linear algebraic equation like

$$ax^2 + bx + c = 0$$

we have

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

In this situation, we have

$$a = w_k^T Q^3 w_k$$
$$b = -2w_k^T Q^2 w_k$$
$$c = w_k^T Q w_k$$

We find that

$$\begin{aligned} b^2 - 4ac &= \left(-2w_k^T Q^2 w_k\right)^2 - 4w_k^T Q^3 w_k w_k^T Q w_k \\ &= 4w_k^T Q^4 w_k - 4w_k^T Q^4 w_k \\ &= 0 \end{aligned}$$

Hence,

$$\begin{aligned} \eta &= \frac{-b}{2a} \\ &= \frac{-\left(-2w_k^T Q^2 w_k\right)}{2w_k^T Q^3 w_k} \\ &= \frac{w_k^T Q^2 w_k}{w_k^T Q^3 w_k} \end{aligned}$$

b) Notice that we have

$$E(w) = 0.5 w^T Q w = 0.5 \begin{bmatrix} w_1 & w_2 \end{bmatrix} Q \begin{bmatrix} w_1 \\ w_2 \end{bmatrix} = 2w_1^2 + 2w_1 w_2 + 5w_2^2$$

Hence

$$Q = \begin{bmatrix} 4 & 2 \\ 2 & 10 \end{bmatrix}$$

From part a) we have

$$\eta = \left. \frac{w^T Q^2 w}{w^T Q^3 w} \right|_{(k)}$$

When $k = 0$, we have

$$\eta = \left.\frac{w^T Q^2 w}{w^T Q^3 w}\right|_{(0)}$$

$$= \frac{\begin{bmatrix}2 & -2\end{bmatrix}\begin{bmatrix}4 & 2\\ 2 & 10\end{bmatrix}^2\begin{bmatrix}2\\ -2\end{bmatrix}}{\begin{bmatrix}2 & -2\end{bmatrix}\begin{bmatrix}4 & 2\\ 2 & 10\end{bmatrix}^3\begin{bmatrix}2\\ -2\end{bmatrix}}$$

$$= \frac{272}{2368} \approx 0.11$$

Combined with $w_{k+1} = w_k - \eta Q w_k$, we have

$$w_1 = w_0 - \eta Q w_0$$

$$= w_0 - \eta Q w_0$$

$$= \begin{bmatrix}2\\ -2\end{bmatrix} - \frac{272}{2368}\begin{bmatrix}4 & 2\\ 2 & 10\end{bmatrix}\begin{bmatrix}2\\ -2\end{bmatrix}$$

$$= \begin{bmatrix}\frac{57}{37}\\ \frac{-6}{37}\end{bmatrix} \approx \begin{bmatrix}1.54\\ -0.16\end{bmatrix}$$

For the gradients, we have

$$\nabla E\left(w(0)\right) = \left.\frac{\partial E(w)}{\partial w}\right|_{(0)} = Qw(0) = \begin{bmatrix}4 & 2\\ 2 & 10\end{bmatrix}\begin{bmatrix}2\\ -2\end{bmatrix} = \begin{bmatrix}4\\ -16\end{bmatrix}$$

$$\nabla E\left(w(1)\right) = \left.\frac{\partial E(w)}{\partial w}\right|_{(1)} = Qw(1) = \begin{bmatrix}4 & 2\\ 2 & 10\end{bmatrix}\begin{bmatrix}1.54\\ -0.16\end{bmatrix} = \begin{bmatrix}5.84\\ 1.48\end{bmatrix}$$

Since $\nabla E\left(w(0)\right)^T * \nabla E\left(w(1)\right) = -0.32 \approx 0$, they are perpendicular.

Solution of Problem 3

a) Set up two sets of input data, each of which is non-decreasing in sequence.
First, get random none-repeating integer by using randperm(1001).
Second, pick 100 from those integers and sort in ascend direction.
Third, map to the interval [-1,1] by linear method.
Then we get the output data by calculating these input data through function y.

b) I write two RBF networks in different Membership Functions but both with K-mean method. However, my k-mean algorithm has flaw so that once the number of nodes exceeds 50, it will not stop clustering. Therefore, I skipped the cluster section for 100 nodes and loosed the cluster standard at 60, 70, 80 and 90 nodes. So data for 60 to 90 is meaningless when compared with.
In my RBF networks, centers would be generated by K-mean method and the Spreads are 3 times the minimum distance between centers.

(1) Different numbers of nodes

For Gaussian Function

Nodes	10	20	30	40	50	60	70	80	90	100
MSE	0.2650	0.1825	0.0094	0.0514	0.0960	6.2e+16	2.5e+16	3.0e+16	2.7e+16	3.3598

For Logistic Function

Nodes	10	20	30	40	50	60	70	80	90	100
MSE	0.2228	0.2764	0.0854	0.0818	0.1374	6.5e+12	7.3e+12	5.0e+22	5.0e+22	0.5399

Obviously, 10 nodes are not enough for the network. As the number of nodes increased, the network performed better. But when it came to 50 nodes, the network got worse. They began to 'remember' the training data.

(2) Different basis functions

(2.1)Comparison in Mean Squared Errors for both networks.

From tables in (1), we can read that Gaussian always gave less error. Below are outcomes from Gaussian and logistic

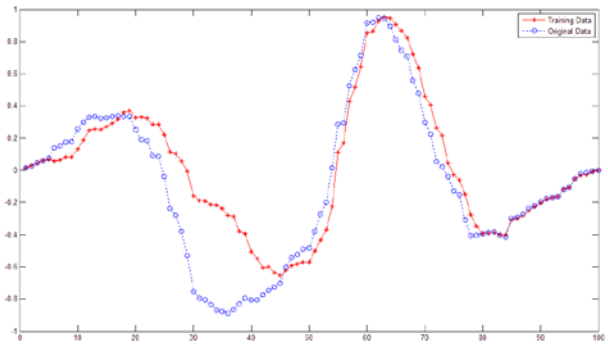


Fig 1. Outcome from Gaussian function network, 40 nodes

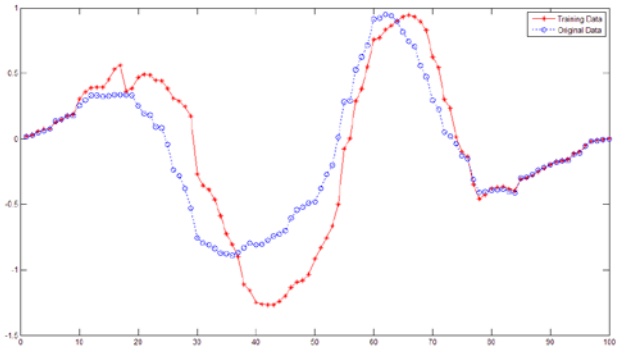


Fig 2. Outcome from logistic function network, 40 nodes

(2.2)Comparison in the execution time for both networks:

For Gaussian Function

Nodes	10	20	30	40	50	60	70	80	90	100
Time	0.3567	0.6960	1.0182	1.4218	1.7776	2.1170	2.4672	2.8060	3.1768	3.5366

For Logistic Function

Nodes	10	20	30	40	50	60	70	80	90	100
time	0.3524	0.6506	0.9670	1.2868	1.6415	1.9157	2.2654	2.5893	2.8861	3.2180

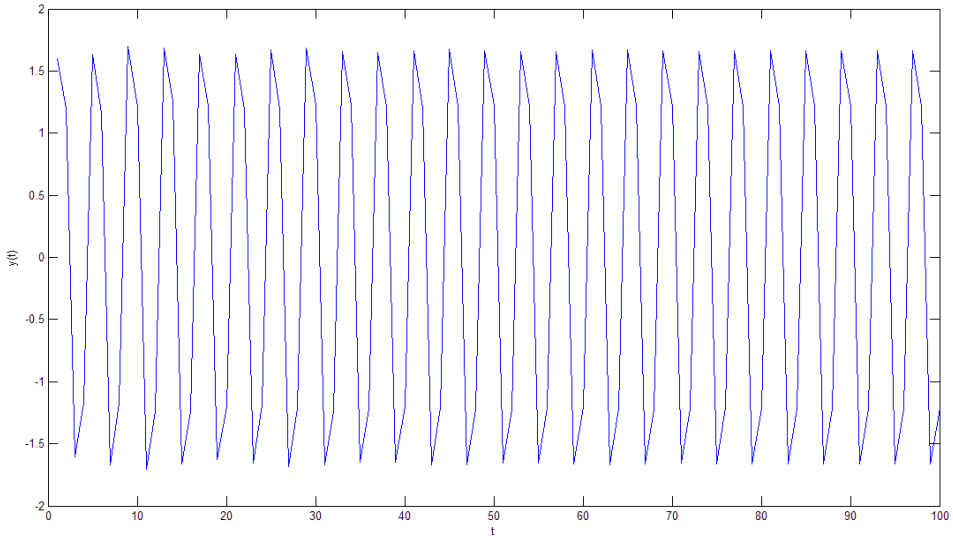
Above tables show that logistic function is faster than Gaussian function.

(3) Discuss

Logistic RBF network learns faster but gives more error. Gaussian RBF network is slow in speed but good in accuracy. This is because their shapes are different, logistic function is 'wider' than Gaussian function. The common parts with functions for other centers are larger, hence faster in speed.

Solution of Problem 4

a) Once we pick y(0) and y(1), we could generate the time series with them. Here, random value is used for y(0) and y(1). I observed that 1.6, 1.2, -1.6,-1.2 showed up in period for some trials. This is the most stable time series for my experiments.



b) Since the time series got 'warmed up' so quickly with my initial conditions, there is no need to discard sample for transient effects.
Root mean squared error is used to measure the network. Hidden layer sizes are denoted as i, j and k, in direction from output to input.
Since the network changed every time I trained it, the result may not represent the real trend. Multiple trials should be done to determine the best network.
For 1 hidden layer network:

	i=2	i=10	i=18	i=26	i=34	i=42	i=50	i=58
RMSE	0.0016	0.0025	0.0142	0.0035	0.0033	0.0046	0.0222	0.0008

The minimum for 1 layers network is 0.0008.

For 2 hidden layers network:

RMSE	j=2	j=10	j=18	j=26	j=34	j=42	j=50	j=58
i=2	0.0008	0.0003	0.0004	0.0016	0.0054	0.0007	0.0006	0.0004
i=10	0.0015	0.0014	0.0017	0.0006	0.0026	0.0004	0.0005	0.0006
i=18	0.0008	0.0004	0.0004	0.0003	0.0046	0.0004	0.0003	0.0084
i=26	0.0007	0.0027	0.0053	0.0001	0.0049	0.0003	0.0002	0.0015
i=34	0.1597	0.0005	0.0004	0.0008	0.0004	0.0003	0.0023	0.0008
i=42	0.0026	0.0003	0.0227	0.0162	0.0002	0.0006	0.0007	0.0051

i=50	0.0006	0.0003	0.0004	0.0194	0.0096	0.0003	0.0011	0.0002
i=58	0.0039	0.0015	0.0022	0.0006	0.0027	0.0114	0.0002	0.0004

The minimum for 2 layers network is 0.0001. Average is 0.0049.

For 3 hidden layers network:

k=2

RMSE	j=2	j=10	j=18	j=26	j=34	j=42	j=50	j=58
i=2	0.000339	0.00024	0.001278	1.455918	0.001176	0.00044	0.00022	0.000418
i=10	0.000402	0.000245	0.002017	0.001051	0.000214	0.000409	0.000888	0.000583
i=18	0.000413	0.00145	0.0008	0.000405	0.000326	0.000342	0.002749	0.000208
i=26	1.862494	0.001023	0.000586	0.002295	0.00054	0.002256	0.004965	0.00386
i=34	0.157712	0.000556	0.002132	0.000926	0.001772	0.000158	0.00515	0.000503
i=42	0.000397	0.00262	0.008571	0.000481	0.000376	0.000122	0.000354	0.002717
i=50	0.001402	0.00047	0.000275	0.000949	0.000211	0.001751	0.000435	0.000287
i=58	0.000673	0.000229	0.000127	0.003227	0.002065	0.003803	0.001523	0.001062

k=10

RMSE	j=2	j=10	j=18	j=26	j=34	j=42	j=50	j=58
i=2	0.000411	2.78844	0.000185	0.000343	0.066182	0.159928	0.000566	0.000339
i=10	0.000456	0.000283	0.00092	0.008191	0.000409	0.002598	0.00045	0.004594
i=18	0.000484	0.000308	0.000697	0.000141	0.000374	0.000806	0.000642	0.00023
i=26	0.000504	0.000315	0.00067	0.000666	0.000302	0.00037	0.011081	0.000524
i=34	0.000545	0.000348	0.000357	0.000306	0.000453	0.000342	0.000174	0.003226
i=42	0.000829	0.000388	0.000464	0.00037	0.000589	0.000639	0.000348	0.002392
i=50	0.159656	0.00026	0.00092	0.001612	0.000332	0.000304	0.000213	0.000302
i=58	0.009069	0.000651	0.000222	0.012333	0.001352	0.000394	0.002412	0.000925

k=18

RMSE	j=2	j=10	j=18	j=26	j=34	j=42	j=50	j=58
i=2	0.000612	0.000248	0.001067	0.00059	0.002939	0.000474	0.00024	0.000557
i=10	0.000801	0.00029	0.000283	0.000238	0.000287	0.00052	0.000412	0.000303
i=18	0.00058	0.001958	0.000933	0.000354	0.000439	0.00048	0.000291	0.000408
i=26	0.158111	0.000301	0.005518	0.000399	0.000795	0.00131	0.000357	0.000413
i=34	0.000781	0.000251	0.000545	0.001134	0.000963	0.000222	0.000469	0.000455
i=42	0.001173	0.000329	0.000316	0.00989	0.000248	0.001615	0.004477	0.002053
i=50	0.003667	0.000647	0.000281	0.001482	0.001931	0.00183	0.000456	0.000192
i=58	0.001187	0.000243	0.000932	0.000913	0.001029	0.000975	0.000123	0.000366

k=26

RMSE	j=2	j=10	j=18	j=26	j=34	j=42	j=50	j=58
i=2	0.001329	0.000185	0.00438	0.000293	0.006723	0.009627	0.000177	0.000463
i=10	0.000235	0.000248	0.00027	0.000243	0.002036	0.002167	0.000852	0.010167
i=18	0.000845	0.000883	0.000493	0.000631	0.000776	0.004652	0.052973	0.06782
i=26	0.000463	0.001124	0.000289	0.001093	0.00313	0.007944	0.011396	0.000935
i=34	0.000368	0.002003	0.000272	0.000234	0.000646	0.02392	7.17E-05	0.034996
i=42	0.000362	0.000539	0.000321	0.004678	0.034063	0.001354	0.000459	0.033807
i=50	0.000417	0.00361	0.000999	0.000901	0.002012	0.000457	0.000582	0.000169
i=58	0.157591	0.0003	0.021285	0.000301	0.001768	0.024782	0.001219	0.000123

k=34

RMSE	j=2	j=10	j=18	j=26	j=34	j=42	j=50	j=58
i=2	0.000389	0.000642	0.000567	0.010228	0.000444	0.000271	0.000144	0.000101
i=10	0.000354	0.000735	0.000962	0.001131	0.005187	0.000965	0.000331	0.004829
i=18	0.000501	0.000375	0.000328	0.000313	0.102533	0.000587	0.000421	0.000367
i=26	0.000322	0.003011	0.000368	0.001795	0.000156	0.000825	0.000303	0.000381
i=34	0.000997	0.029574	0.000414	0.000486	0.000351	0.004787	0.003252	0.000358
i=42	0.0002	0.000724	0.001879	0.00032	0.000554	0.001111	0.000233	0.000236
i=50	0.000458	0.009691	0.002585	0.010974	0.00417	0.000371	0.014059	0.001681
i=58	0.005707	0.000414	0.000458	0.000168	0.000391	0.000575	0.001712	0.000358

k=42

RMSE	j=2	j=10	j=18	j=26	j=34	j=42	j=50	j=58
i=2	0.000458	0.009691	0.002585	0.010974	0.00417	0.000371	0.014059	0.001681
i=10	0.005707	0.000414	0.000458	0.000168	0.000391	0.000575	0.001712	0.000358
i=18	0.000358	0.001155	0.000425	0.004989	0.00016	0.000815	0.008739	0.00052
i=26	0.000651	0.000887	0.00027	0.000288	0.000894	0.000143	0.003696	0.00232
i=34	0.00083	0.000293	0.001683	0.001007	0.00261	0.000431	0.000188	0.001947
i=42	0.000398	0.001987	0.000226	0.000968	0.013081	0.000689	0.000365	0.000338
i=50	0.000423	0.002783	0.00401	0.000461	0.000316	0.01073	0.000535	0.001194
i=58	0.000338	0.002215	0.000366	0.000372	0.007232	0.004491	0.018351	0.007192

k=50

RMSE	j=2	j=10	j=18	j=26	j=34	j=42	j=50	j=58
i=2	0.22501	0.002807	0.000601	0.015442	0.000421	0.000474	0.010558	0.000368
i=10	0.000354	0.000123	0.009963	0.008566	0.001615	0.00332	0.010952	0.000421
i=18	0.001565	0.003579	0.005382	0.002544	0.002017	0.000402	0.001772	0.0002
i=26	0.00063	0.000146	0.000273	0.001312	0.003307	0.015574	0.073239	0.000298
i=34	0.157599	0.008913	0.001045	0.001507	0.000432	0.008936	0.003119	0.000249
i=42	0.000441	0.000377	0.001471	0.000226	0.012559	0.000368	0.004395	0.000945
i=50	0.000482	0.002478	0.000375	0.000879	0.109051	0.002323	0.002371	0.000385
i=58	0.006009	0.002381	0.002972	0.002292	0.000325	0.002501	0.000613	0.001783

k=58

RMSE	j=2	j=10	j=18	j=26	j=34	j=42	j=50	j=58
i=2	0.000344	0.000206	0.047062	0.005973	0.000641	0.001047	0.003265	0.00617
i=10	0.00036	0.003043	0.001083	0.00042	0.000407	0.020726	0.000731	0.021263
i=18	0.001916	0.001598	0.000262	0.000343	0.002872	0.000181	0.000236	0.001896
i=26	0.000946	0.000111	0.000479	0.004138	0.006967	0.126026	0.002327	0.051453
i=34	0.157552	0.000273	0.004571	0.017063	0.000228	0.003448	0.003006	0.001668
i=42	0.001008	0.000373	0.045277	0.001111	0.001668	0.000496	0.00138	0.007268
i=50	0.000607	0.01584	0.065536	0.02714	0.052322	0.03708	0.034092	0.042987
i=58	0.131239	0.154462	0.012487	0.005503	0.016098	0.088185	0.185142	0.051156

The minimum for 3 layers network is 0.0007.

So the best net work should be 2 hidden layers network with 26 nodes for each layer.

c) Comparison with RBF networks showed that RBF networks were more efficient at the task. RBF is much faster and stable.

	RMSE

feed forward network	0.0049
RBF 30	0.00023730
RBF 60	0.00022056
RBF 90	0.00014820

Solution of Problem 5

5.1)I used the Matlab neural network toolbox to create the classifier. A two hidden layers feed forward network was chosen. There are four key parameters namely hidden layer1 size, hidden layer2 size, learning rate and goal. I took mean squared errors, error numbers in classification and numbers of used epochs as reference parameters.

hidden layer1 size	hidden layer2 size	learning rate	goal	mse for testdata	errors	epochs used
10	10	0.1	0.001	0.0104	0	6
10	10	0.1	0.001	0.0199	1	6
10	10	0.1	0.001	0.0978	3	6
10	10	0.1	0.001	0.0473	0	7
10	10	0.1	0.001	0.0555	2	8
10	10	0.1	0.001	0.0486	0	7
10	10	0.1	0.001	0.0117	0	7
10	10	0.1	0.001	0.0617	2	5
10	10	0.1	0.001	0.0675	2	7
10	10	0.1	0.001	0.1166	2	8
10	10	0.5	0.001	0.0348	0	6
10	10	0.5	0.001	0.0305	2	8
10	10	0.5	0.001	0.0293	0	7
10	10	0.5	0.001	0.0992	1	7
10	10	0.5	0.001	0.0731	1	5
10	10	0.5	0.001	0.0939	0	7
10	10	0.5	0.001	0.0987	1	7
10	10	0.5	0.001	0.0755	2	8
10	10	0.5	0.001	0.0728	1	5
10	10	0.5	0.001	0.046	1	7
10	10	0.9	0.001	0.0409	0	6
10	10	0.9	0.001	0.0269	1	5
10	10	0.9	0.001	0.0311	0	10
10	10	0.9	0.001	0.0646	4	7
10	10	0.9	0.001	0.1607	1	7
10	10	0.9	0.001	0.0684	1	7
10	10	0.9	0.001	0.0585	1	10
10	10	0.9	0.001	0.0683	2	7
10	10	0.9	0.001	0.0488	2	6
10	10	0.9	0.001	0.0755	1	6
10	10	0.1	0.0001	0.0225	1	7
10	10	0.1	0.0001	0.1209	2	9

10	10	0.1	0.0001	0.0541	1	8
10	10	0.1	0.0001	0.041	0	10
10	10	0.1	0.0001	0.0414	0	9
10	10	0.1	0.0001	0.0687	3	8
10	10	0.1	0.0001	0.0466	1	8
10	10	0.1	0.0001	0.0322	3	10
10	10	0.1	0.0001	0.0744	2	9
10	10	0.1	0.0001	0.0631	3	7
10	10	0.1	0.01	0.0585	1	5
10	10	0.1	0.01	0.0721	0	9
10	10	0.1	0.01	0.06	1	10
10	10	0.1	0.01	0.0695	0	4
10	10	0.1	0.01	0.0889	0	5
10	10	0.1	0.01	0.0937	2	10
10	10	0.1	0.01	0.0506	0	6
10	10	0.1	0.01	0.0421	1	8
10	10	0.1	0.01	0.131	1	8
10	10	0.1	0.01	0.0489	0	5
5	10	0.1	0.001	0.0552	1	6
5	10	0.1	0.001	0.0474	1	9
5	10	0.1	0.001	0.0309	2	5
5	10	0.1	0.001	0.0977	1	5
5	10	0.1	0.001	0.0726	5	6
5	10	0.1	0.001	0.0907	2	9
5	10	0.1	0.001	0.0688	1	9
5	10	0.1	0.001	0.1365	4	6
5	10	0.1	0.001	0.0377	3	7
5	10	0.1	0.001	0.1395	5	5
10	5	0.1	0.001	0.1379	4	6
10	5	0.1	0.001	0.1098	2	4
10	5	0.1	0.001	0.1156	1	7
10	5	0.1	0.001	0.0722	0	7
10	5	0.1	0.001	0.0746	0	5
10	5	0.1	0.001	0.0509	1	4
10	5	0.1	0.001	0.1771	4	8
10	5	0.1	0.001	0.0527	1	10
10	5	0.1	0.001	0.0832	1	5
10	5	0.1	0.001	0.0589	1	6
20	20	0.1	0.001	0.0733	1	4
20	20	0.1	0.001	0.1508	3	3
20	20	0.1	0.001	0.0432	0	5
20	20	0.1	0.001	0.1324	2	3
20	20	0.1	0.001	0.1053	1	4
20	20	0.1	0.001	0.0872	0	2
20	20	0.1	0.001	0.0544	2	3

20	20	0. 1	0. 001	0. 1184	1	4
20	20	0. 1	0. 001	0. 0617	1	4
20	20	0. 1	0. 001	0. 043	1	4

a) Learning rate: 0.1 is better than 0.5 and 0.9.
b) Goal does: 0.01 is enough.
c) Hidden layer size: 10 for each layer

5.2)The result is:

a belongs to product 1
b belongs to product 2
c belongs to product 3