Name: Zhihan Wang

Student ID: 17272381

Section 1: Multi-Layer Perception:

-perform any pre-processing of the data;

In this section, all its source code is in 'section 1' file.

There are total 58 features given by the Spambase dataset as per description, the last feature denotes whether the e-mail was considered spam (1) or not (0).

The first 57 features has been normalised by dividing maximum.

-implement two class output encoding for the MLP;

The last feature has been encoded as target dataset that convert the targets into 1-of-N encoding. [0,1] for ham email and [1,0] for spam email.

Then input dataset is sliced into training, testing and validation set with 50:25:25 ratio. The target dataset has the same arrangement as training_target, testing_target and validation_target set with 50:25:25 ratio and same order as input dataset.

-train a simple MLP (choose a number of hidden nodes that seems reasonable) and see how well you can perform the classification. Use the confusion matrix to output the results;

mlp class provided by this course has been used to compute multi-layer perception (MLP) task**.** Some of the parameter for this MLP model has been fixed to test performance in this assignment, which are learning rate is set up to 0.2, momentum is 0.9, beta is 1.0, 'softmax' has been used as activation function.

Initially, 5 hidden nodes has been chosen to test the outcome. The 1-57 features of spam data set is used as input of MLP model, 'mlp.earlystopping' function has been used to avoid overfitting problem.

The result shows the percentage of correct is round about 92-93%.

- implement cross-validation;

Cross validation has been applied with the same set up of MLP network parameters (5 hidden nodes, 0.2 of learning rate , 0.9 of momentum , 1.0 of beta , 'softmax' of activation function) as above.

Cross validation 1 data set has 50% head for training, 25% middle for validation, and 25% tail for testing.

Cross validation 2 data set has 50% middle for training, 25% tail for validation, and 25% head for testing.

The overall result for above shows below, the output of MLP training is similar:

	Input					**Output**		
Test Name	**no. of hidden nodes**	**features**	**learning rate**	**momentum**	**activation function**	**iterations x 100**	**errors**	**Percentage Correct**
first MLP	5	1 to 57	0.2	0.9	softmax	233	61.92	92.35
cross validation 1	5	1 to 57	0.2	0.9	softmax	500	61.22	93.22
cross validation 2	5	1 to 57	0.2	0.9	softmax	312	75.76	92.79

-test out different sizes of hidden layer to see how many hidden neurons give the best results;

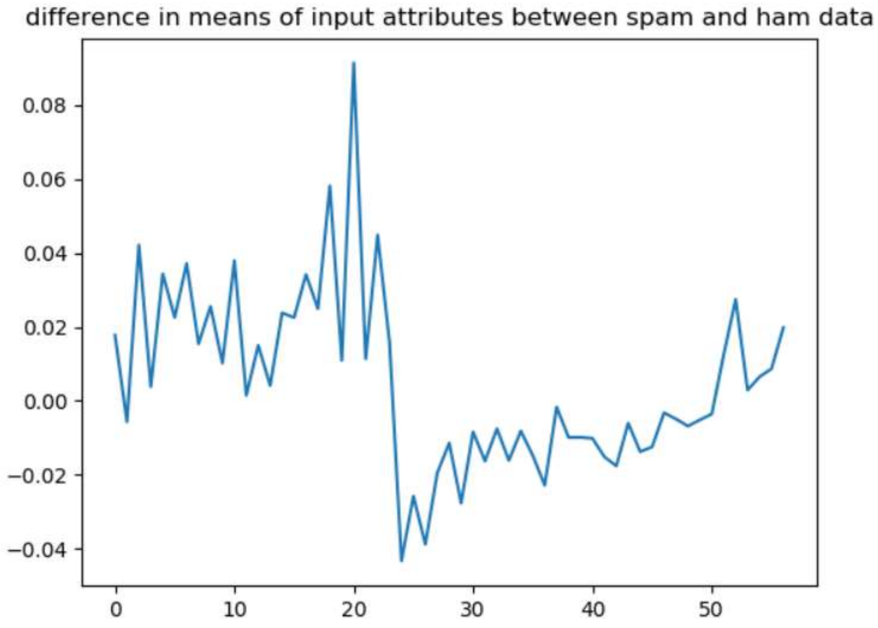
The mlp class provided by course shows there is one hidden layer for this MLP model. Therefore, different number of hidden nodes has been tested with the same parameter MLP network with same input and output data structure for comparison.

The result shows performance are similar for different hidden nodes of MLP network, 1 and 5 hidden nodes of MLP network gives the slightly better performance. Therefore, 1 hidden node will be used for MLP network in the following tests as it is the simplest model.

	Input					**Output**		
Test Name	**no. of hidden nodes**	**features**	**learning rate**	**momentum**	**activation function**	**iterations x 100**	**errors**	**Percentage Correct**
1 hidden node	1	1 to 57	0.2	0.9	softmax	258	61.96	92.43
2 hidden nodes	2	1 to 57	0.2	0.9	softmax	244	61.86	92.35
5 hidden nodes	5	1 to 57	0.2	0.9	softmax	257	61.78	92.43
10 hidden nodes	10	1 to 57	0.2	0.9	softmax	232	61.91	92.35
20 hidden nodes	20	1 to 57	0.2	0.9	softmax	237	61.93	92.35
25 hidden nodes	25	1 to 57	0.2	0.9	softmax	243	61.87	92.35
30 hidden nodes	30	1 to 57	0.2	0.9	softmax	241	61.87	92.35
35 hidden nodes	35	1 to 57	0.2	0.9	softmax	238	61.89	92.35
40 hidden nodes	40	1 to 57	0.2	0.9	softmax	236	61.94	92.35
45 hidden nodes	45	1 to 57	0.2	0.9	softmax	235	61.93	92.35
50 hidden nodes	50	1 to 57	0.2	0.9	softmax	238	61.92	92.35

-test out different subsets of the features to see which are useful;

Difference in means of input attributes between spam and ham data shows 1-23 features has higher means for spam email data and 23-57 features has higher means for ham email data



Therefore, features has been separated into 7 sets: 1-23 features, 24-57 features, even number of all features , odd number of all features, 1-48 features for word_freq_WORD , 49 – 54 features for char_freq_CHAR and 55-57 features for sum of data attributes from spam data for comparison,

The result shows in table below.

24-57 features set shows the best result, however, it has longer training time .

	Input					Output		
Test Name	no. of hidden nodes	features	learning rate	momentum	activation function	iterations x 100	errors	Percentage Correct
sub features 1	1	1 to 23	0.2	0.9	softmax	803	112.43	84.26
sub features 2	1	24 to 57	0.2	0.9	softmax	1205	92.02	90.17
sub features 3	1	even number of 1-57 features	0.2	0.9	softmax	2575	81.72	90.09
sub features 4	1	odd number of 1-57 features	0.2	0.9	softmax	677	94.13	88.43
sub features 5	1	1 to 48	0.2	0.9	softmax	199	81.27	89.74
sub features 6	1	49 to 54	0.2	0.9	softmax	284	139.41	84.17
sub features 7	1	55 to 57	0.2	0.9	softmax	1350	210.51	76.60

In short, more features input gives better performance for MLP network training.

All the above output results in table in this section could have variance since data has been shuffled at beginning that each time when running the programme.

Section 2: MLP trained using GA

-encode the weights of a neural network in a string representation;

In this section, its source code is in 'section 2' file and 'fitness_function' file

Each single weight has been encoded as format of a 16 bit binary code:

i.e. [0, 1, 1011, 1101111111], then it could be transferred from binary to decimal.

The first digit is represented as connectivity/edge for each weight towards its target layer, which is hidden layer or output layer, 0 for not connected and 1 for connected.

The second digit is used as sign to compute +- of a decimal number.

The 3rd – 6th digit are represented as integer part of decimal with a range of 0-16 decimal number. The 7th – 16th digit are represented as floating part of this decimal number.

-code the fitness function to evaluate the fitness of the chromosome representing the weights of the MLP

Two versions of fitness function has been developed in this section

Version 1: Percent of correct has been set as benchmark for the return of fitness function as an input of GA algorithm. Higher percent of correct, better quality of population for evolving.

Version 2: fitness function would be combination for percent of correct and complexity of the MLP network. The idea is less edges for the connection between input layer and hidden, and between hidden layer and output layer, means the MLP network with the same performance has less complexity for saving of computational cost. The fitness function is sum of (percent of correct and 500 / (edges + 1)), higher score gives better result for evolving.

- test out the algorithm and compare the classification accuracy with the MLP trained using back propagation

GA class provided by this course has been used to compute the graph for the performance of GA. 1 hidden node MLP network and 1-57 features of data has been selected for this job.

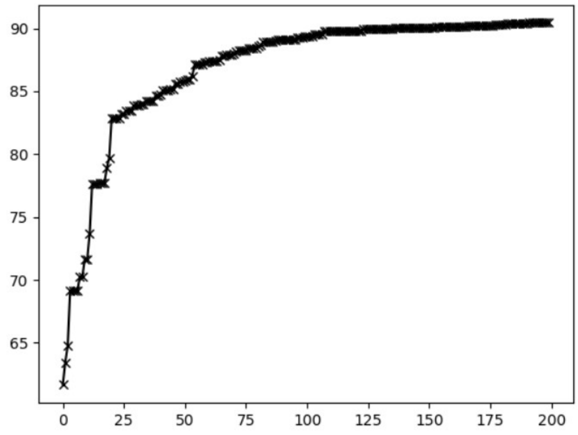
The overall number of weights for input layer, hidden layer and output layer to compute encoded string length would be

(Number of hidden nodes * (number of data features + 1) + (number of hidden nodes+ 1) * number of targets) * 16 = (1 * (57+1) + (1+1)* 2) * 16 = 992

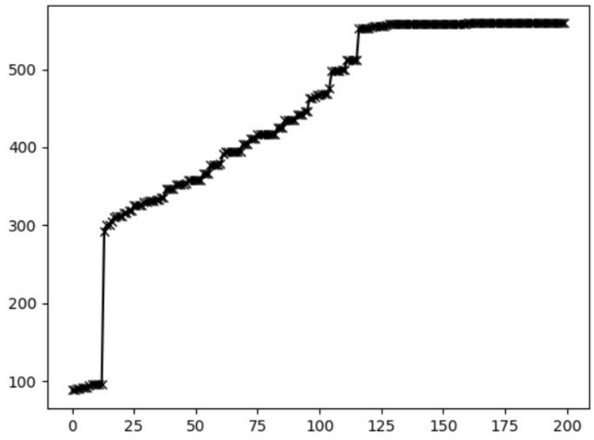
Therefore, it would be a large vector of population size, which is computational expensive. 1 hidden nodes has been selected in this section which is simplest MLP network in section 1.

The nEpochs = 200, population size is 100, mutationProb = -1, single point crossover, nElite =4, tournament = True are the parameters for both version GA algorithm test. Evolution graph shows as below:

Version 1 fitness function:



Version 2 fitness function:



The result for two version of fitness function shows as below:

	Input				Output		
Test Name	**String length**	**fitness function**	**no. of hidden nodes**	**features**	**edges**	**errors**	**Percentage Correct**
GA version 1	992	MLP percentage correct	1	1 to 57	36	208.92	90.04
GA version 2	992	MLP percentage correct + 500/ (number of edges + 1)	1	1 to 57	5	373.30	82.62

The result of 1 hidden node MLP network tested before as below:

	Input					Output		
Test Name	**no. of hidden nodes**	**features**	**learning rate**	**momentum**	**activation function**	**iterations**	**errors**	**Percentage Correct**
1 hidden node	1	1 to 57	0.2	0.9	softmax	258	61.96	92.43

Therefore, the performance GA version 1 is slightly worse than same size MLP network training. However, a less complexity model with 36 edges model has been found compared with MLP network training**.**

The performance GA version 2 has much worse performance, but it has the simplest MLP network, which has only 5 edges.

In GA algorithm of this section, there is no tool to deal with overfitting problem compared with MLP network. It is also computational expensive for a very long string length.

However, MLP BP network has problem for stuck in local minimum that increase its training time significantly.

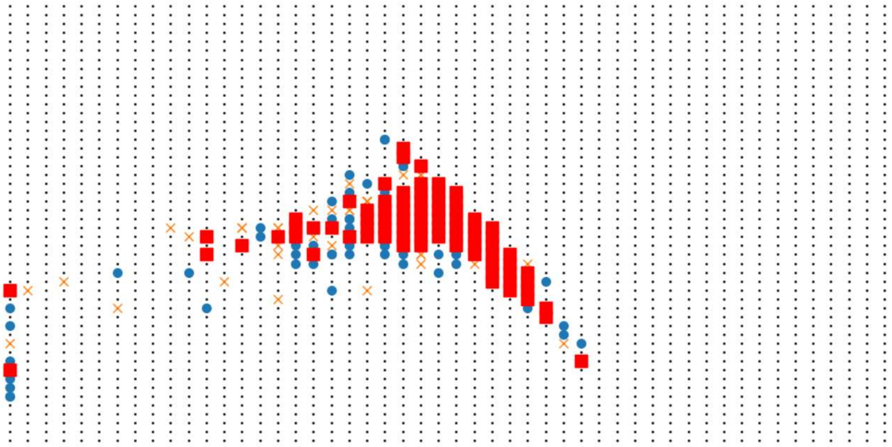
Section 3: Self Organising Map

Source file of this section is in 'section 3' file

- Use the SOM to try to cluster the data and see whether you can identify the spam in the clusters;

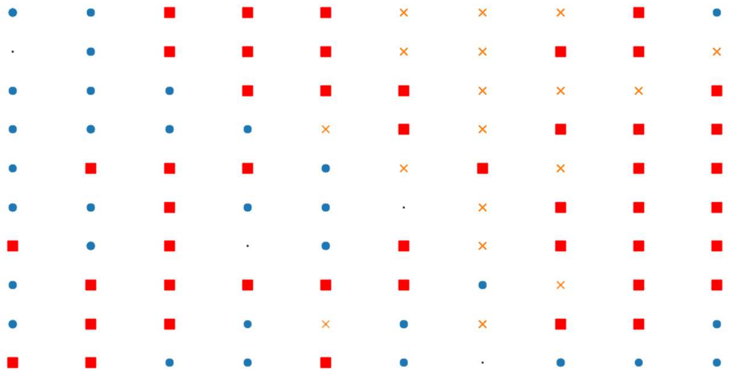
A 50 x 50 SOM network has been used to visualise the data , blue circle is ham email, yellow cross is spam email, black dot is neuron that not fired and red square is overlaps for both ham and spam email. The same legend will be used in the following graphs.

Spam and ham email data almost stack together without training.

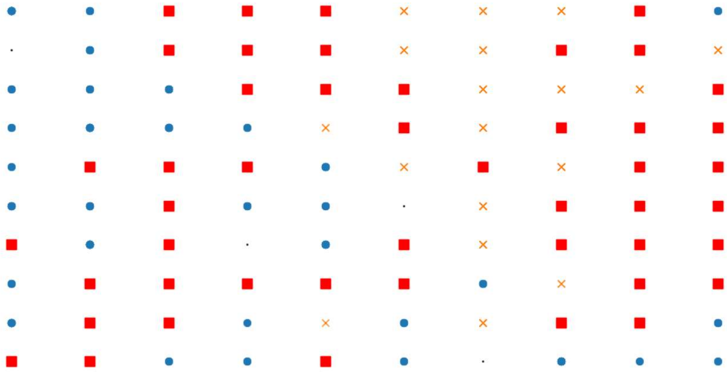


- Use a Perceptron to take the activations of the SOM neurons as input and learn the outputs classes. How well does this work compared to the MLP?

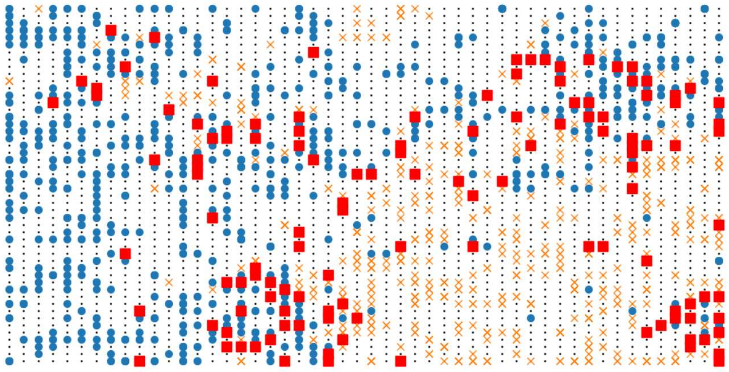
-Train and test all features of spam data as input with parameters of 100 iterations and default set up in som class given by this course:



Graph 2: a 10 x 10 SOM network used, 47 overlap neurons has been found

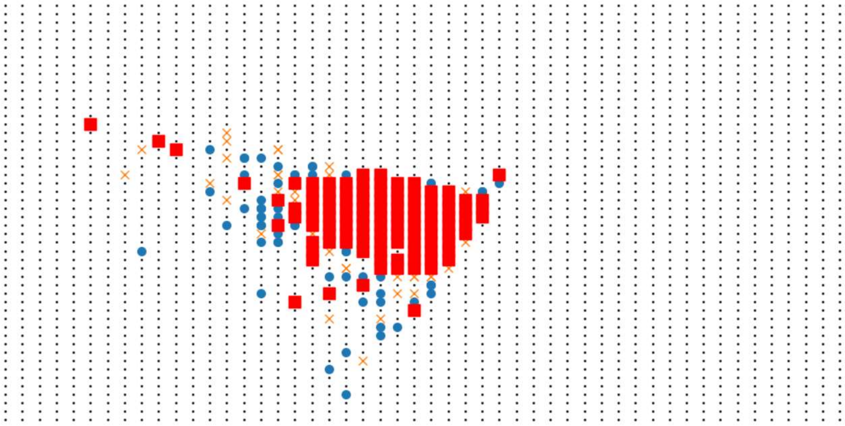


Graph 3: a 20 x 20 SOM network used, 121 overlap neurons has been found

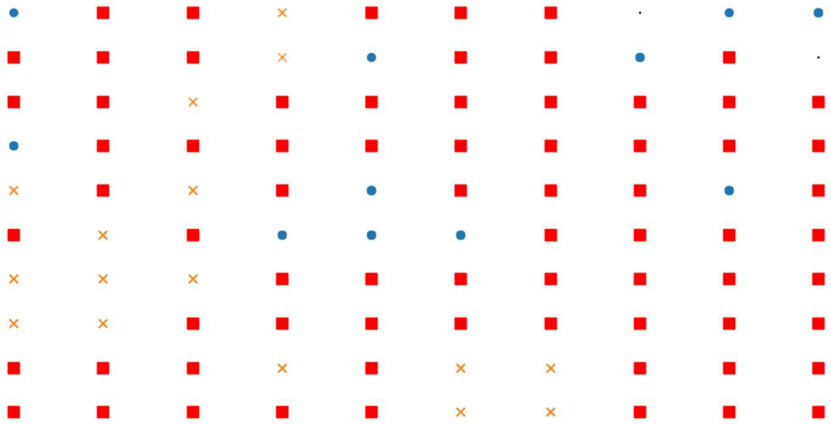


Graph 4: 50 x 50 SOM network used, 125 overlap neurons has been found

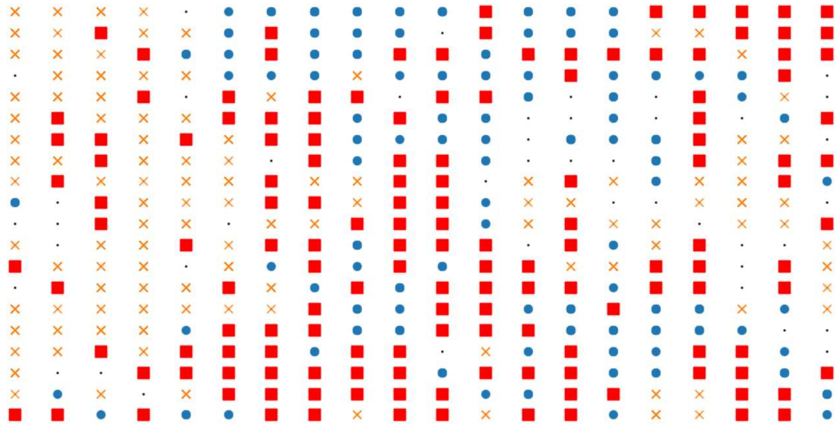
-Train and test 1-23 features of spam data as input with parameters of 100 iterations and default set up in som class given by this course:



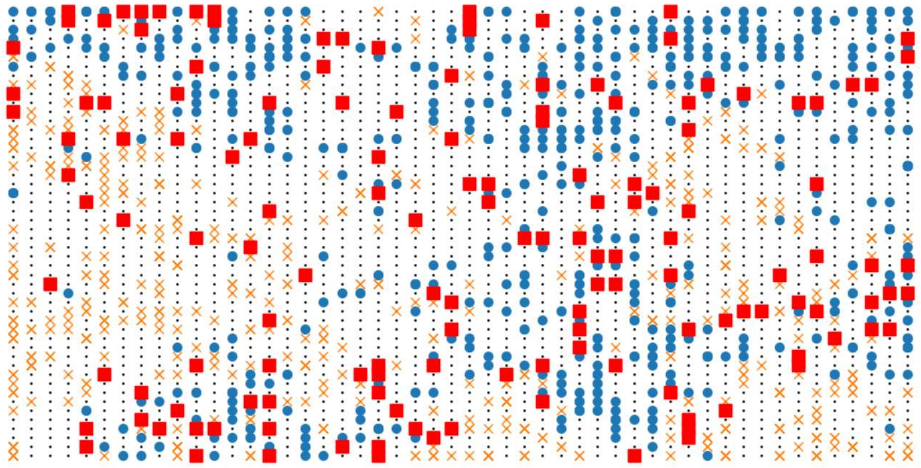
Graph 5: 50 x 50 SOM network used without training



Graph 6: a 10 x 10 SOM network used, 148 overlap neurons has been found

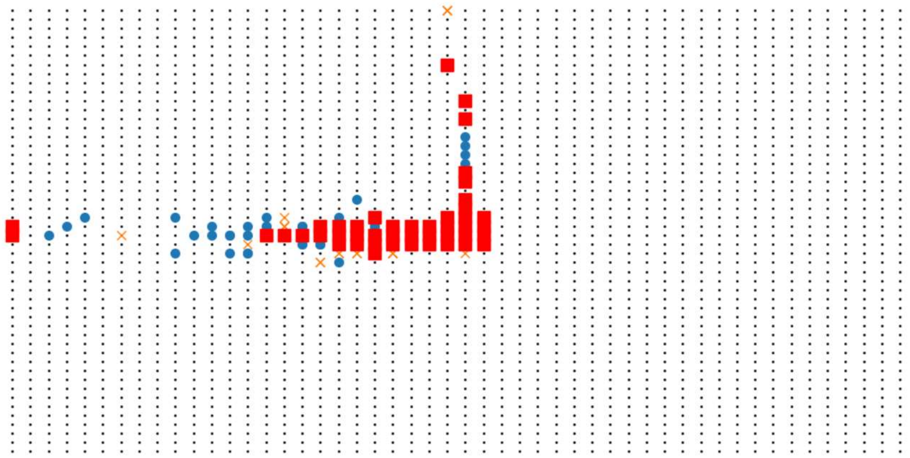


Graph 7: a 20 x 20 SOM network used, 148 overlap neurons has been found

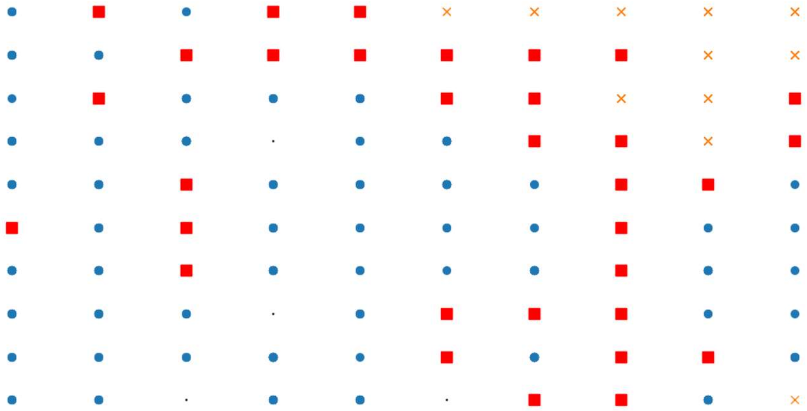


Graph 8: a 50 x 50 SOM network used, 145 overlap neurons has been found

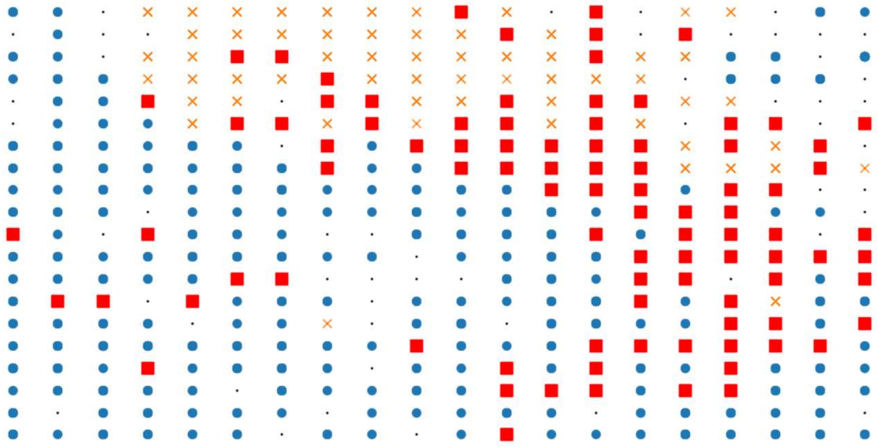
-Train and test 24-57 features of spam data as input with parameters of 100 iterations and default set up in som class given by this course:



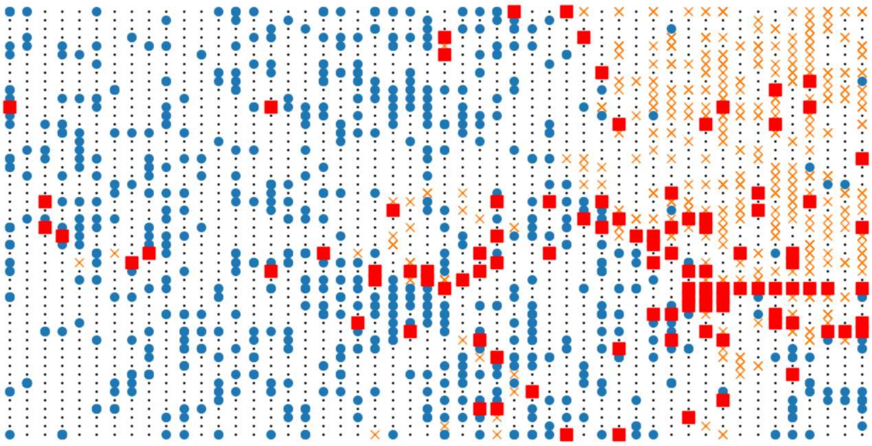
Graph 9: a 50 x 50 SOM network used without training.



Graph 10: a 10 x 10 SOM network used, 32 overlap neurons has been found



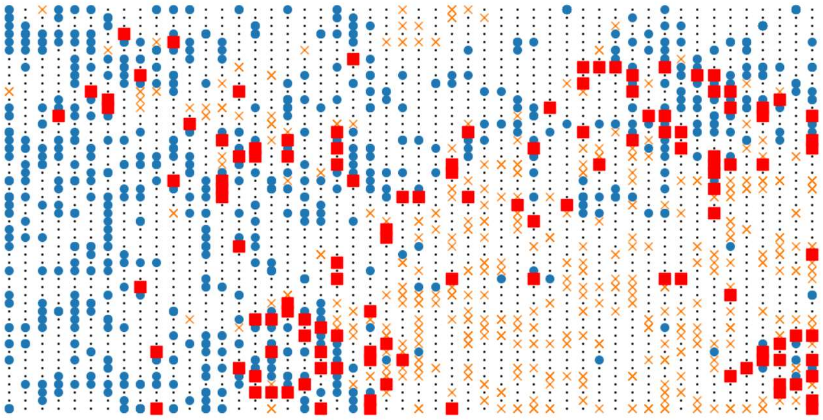
Graph 11: a 20 x 20 SOM network used, 77 overlap neurons has been found



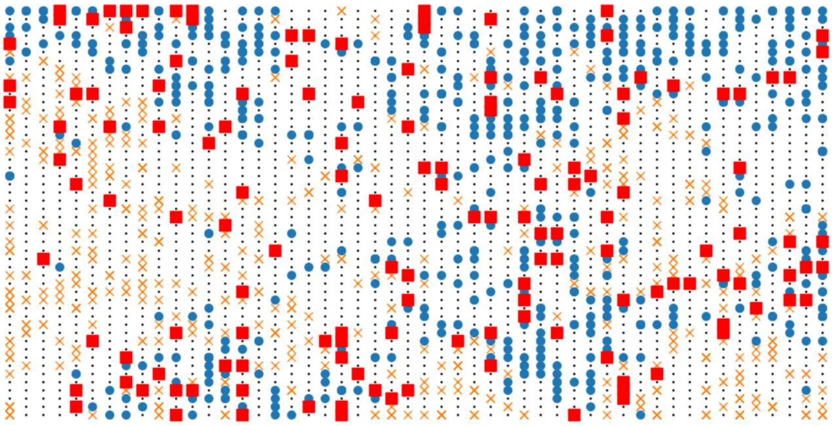
Graph 12: a 50 x 50 SOM network used, 93 overlap neurons has been found

In summary, SOM is an unsupervised learning, the bigger size of SOM network plus more iterations for training, the less proportion of overlaps neurons, which means the different classes of data could be visualised easily. However, the large size of network for training means computational expensive.

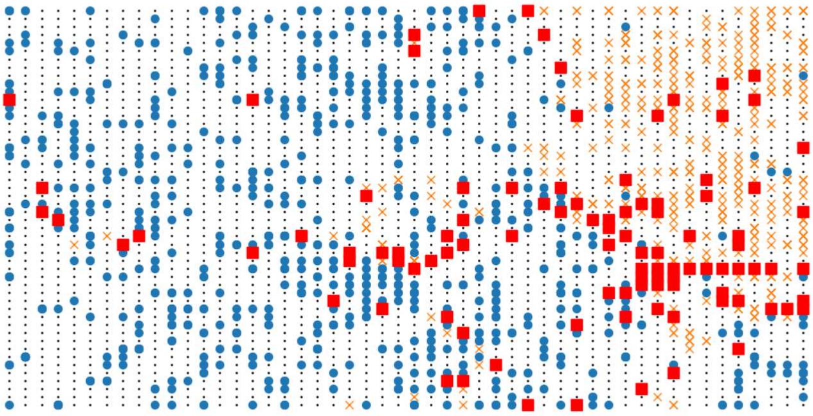
In comparison with different subset of features for a 50 x 50 SOM network, Graph 4, Graph 8 and Graph 12 from previous test has been showed below.



Graph 4



Graph 8



Graph 12

The graph 12 shows a best proportion of 60% spam email and 40% spam email, which using 24-57 spam database features as input.

A disadvantage for SOM training is that there is no easy way to evaluate its performance, while the MLP network has tool such as confusion matrix to evaluate performance and earlystopping function to avoid overfitting problem, in this case MLP network with 24-57 features of database input gives 90.4% accuracy.

Finally, after more than 1 hour training of a 100 x 100 size and 400 iterations SOM with 24-57 features of spam data, the graph of its result as below shows 65 overlaps neurons.