CAP-6619 Deep Learning Fall 2018
Homework 3
Christian Garbin

Question 1 [2 pts]

Table 1 shows a toy dataset with six instances (including two features and one class label), please use t_1=(-1, 0.5) and t_2=(1, -0.5) as the two centers, and use the following RBF kernels to convert the six instances into a new space:

Table 1:

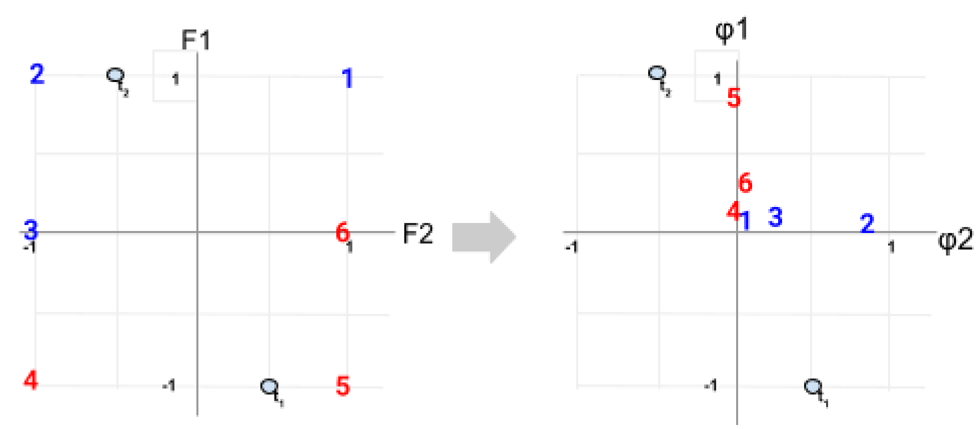
Instance Index	F1	F2	Label
1	1	1	1
2	1	-1	1
3	0	-1	1
4	-1	-1	0
5	-1	1	0
6	0	1	0

Please use Gaussian RBF kernel to (σ=1) convert the six instances into new space. Report the new feature values for each instance.

$$\varphi_1 = exp\Big(\frac{-||x-t_1||^2}{\sigma^2}\Big) = exp\Big(\frac{-||x-t_1||^2}{1^2}\Big) = exp(-||x-t_1||^2) = exp\Big(-(\sqrt{(x_1-t_{11})^2+(x_2-t_{12})^2})^2\Big) = exp(-((x_1-t_{11})^2+(x_2-t_{12})^2))$$

$$\varphi_2 = exp\Big(\frac{-||x-t_2||^2}{\sigma^2}\Big) = exp\Big(\frac{-||x-t_2||^2}{1^2}\Big) = exp(-||x-t_2||^2) = exp\Big(-(\sqrt{(x_1-t_{21})^2+(x_2-t_{22})^2})^2\Big) = exp(-((x_1-t_{21})^2+(x_2-t_{22})^2))$$

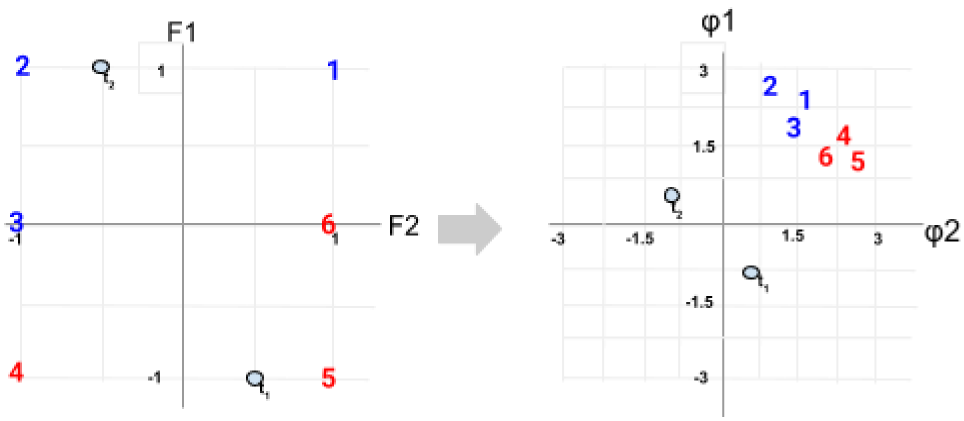
	F_1	F_2	φ_1	φ_2
1	1	1	exp(- || (1,1)-(-1,0.5) || ^{2}) = exp(-((1-(-1))^2 + (1-0.5)^2)) = **0.014**	exp(- || (1,1)-(1,-0.5) || ^{2}) = exp(-((1-1)^2 + (1-(-0.5))^2)) = **0.105**
2	1	-1	exp(- || (1,-1)-(-1,0.5) || ^{2}) = exp(-((1-(-1))^2 + (-1-0.5)^2)) = **0.002**	exp(- || (1,-1)-(1,-0.5) || ^{2}) = exp(-((1-1)^2 + (-1-(-0.5))^2)) = **0.779**
3	0	-1	exp(- || (0,-1)-(-1,0.5) || ^{2}) = exp(-((0-(-1))^2 + (-1-0.5)^2)) = **0.039**	exp(- || (0,-1)-(1,-0.5) || ^{2}) = exp(-((0-1)^2 + (-1-(-0.5))^2)) = **0.287**
4	-1	-1	exp(- || (-1,-1)-(-1,0.5) || ^{2}) = exp(-((-1-(-1))^2 + (-1-0.5)^2)) = **0.105**	exp(- || (-1,-1)-(1,-0.5) || ^{2}) = exp(-((-1-1)^2 + (-1-(-0.5))^2)) = **0.014**
5	-1	1	exp(- || (-1,1)-(-1,0.5) || ^{2}) = exp(-((-1-(-1))^2 + (-0.5)^2)) = **0.779**	exp(- || (-1,1)-(1,-0.5) || ^{2}) = exp(-((-1-1)^2 + (1-(-0.5))^2)) = **0.002**
6	0	1	exp(- || (0,1)-(-1,0.5) || ^{2}) = exp(-((0-(-1))^2 + (1-0.5)^2)) = **0.287**	exp(- || (0,1)-(1,-0.5) || ^{2}) = exp(-((0-1)^2 + (1-(-0.5))^2)) = **0.039**



Please use Multiquadratics RBF kernel to (c=1) convert the six instances into new space. Report the new feature values for each instance.

$$\varphi_1 = \sqrt{||x-t_1||^2+c^2} = \sqrt{(\sqrt{(x_1-t_{11})^2+(x_2-t_{12})^2})^2+1^2} = \sqrt{(x_1-t_{11})^2+(x_2-t_{12})^2+1}$$
$$\varphi_2 = \sqrt{||x-t_2||^2+c^2} = \sqrt{(\sqrt{(x_1-t_{21})^2+(x_2-t_{22})^2})^2+1^2} = \sqrt{(x_1-t_{21})^2+(x_2-t_{22})^2+1}$$

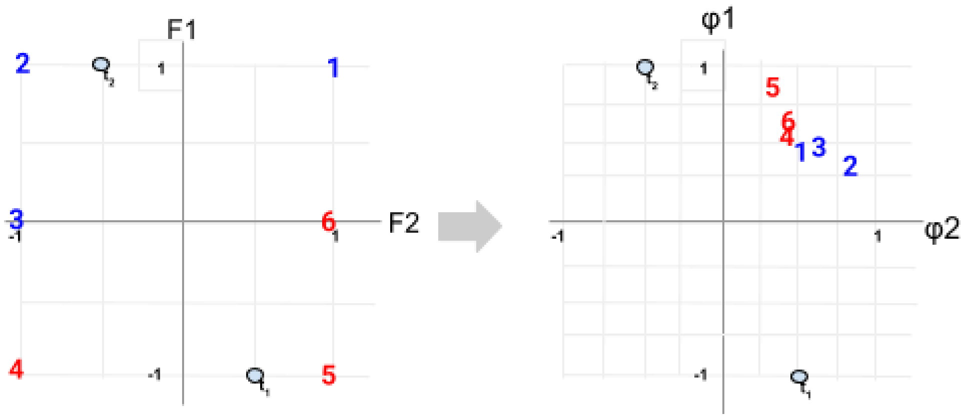
	F_1	F_2	φ_1	φ_2
1	1	1	sqrt(|| (1,1)-(-1,0.5) || ^{2}+1) = sqrt((1-(-1))^2 + (1-0.5)^2+1) = **2.291**	sqrt(|| (1,1)-(1,-0.5) || ^{2}+1) = sqrt((1-1)^2 + (1-(-0.5))^2+1) = **1.803**
2	1	-1	sqrt(|| (1,-1)-(-1,0.5) || ^{2}+1) = sqrt((1-(-1))^2 + (-1-0.5)^2+1) = **2.693**	sqrt(|| (1,-1)-(1,-0.5) || ^{2}+1) = sqrt((1-1)^2 + (-1-(-0.5))^2+1) = **1.118**
3	0	-1	sqrt(|| (0,-1)-(-1,0.5) || ^{2}+1) = sqrt((0-(-1))^2 + (-1-0.5)^2+1) = **2.062**	sqrt(|| (0,-1)-(1,-0.5) || ^{2}+1) = sqrt((0-1)^2 + (-1-(-0.5))^2+1) = **1.500**
4	-1	-1	sqrt(|| (-1,-1)-(-1,0.5) || ^{2}+1) = sqrt((-1-(-1))^2 + (-1-0.5)^2+1) = **1.803**	sqrt(|| (-1,-1)-(1,-0.5) || ^{2}+1) = sqrt((-1-1)^2 + (-1-(-0.5))^2+1) = **2.291**
5	-1	1	sqrt(|| (-1,1)-(-1,0.5) || ^{2}+1) = sqrt((-1-(-1))^2 + (-0.5)^2+1) = **1.118**	sqrt(|| (-1,1)-(1,-0.5) || ^{2}+1) = sqrt((-1-1)^2 + (1-(-0.5))^2+1) = **2.693**
6	0	1	sqrt(|| (0,1)-(-1,0.5) || ^{2}+1) = sqrt((0-(-1))^2 + (1-0.5)^2+1) = **1.500**	sqrt(|| (0,1)-(1,-0.5) || ^{2}+1) = sqrt((0-1)^2 + (1-(-0.5))^2+1) = **2.062**



Please use Inverse Multiquadratics kernel to (c=1) convert the six instances into new space. Report the new feature values for each instance.

$$\varphi_1 = \left(\sqrt{||x-t_1||^2+c^2}\right)^{-1} = \left(\sqrt{(\sqrt{(x_1-t_{11})^2+(x_2-t_{12})^2})^2+1^2}\right)^{-1} = \left(\sqrt{(x_1-t_{11})^2+(x_2-t_{12})^2+1}\right)^{-1}$$
$$\varphi_2 = \left(\sqrt{||x-t_2||^2+c^2}\right)^{-1} = \left(\sqrt{(\sqrt{(x_1-t_{21})^2+(x_2-t_{22})^2})^2+1^2}\right)^{-1} = \left(\sqrt{(x_1-t_{21})^2+(x_2-t_{22})^2+1}\right)^{-1}$$

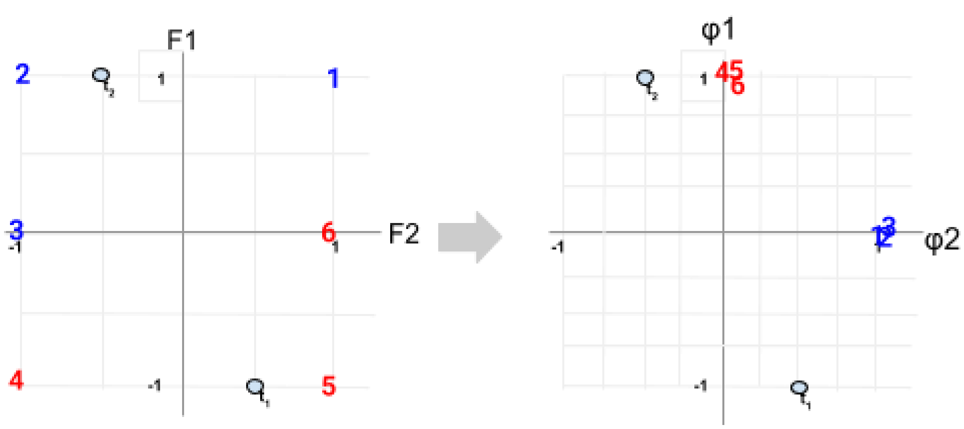
	F_1	F_2	φ_1	φ_2
1	1	1	1/√(|| (1,1)-(-1,0.5) || ^{2}+1) = 1/√((1-(-1))^2 + (1-0.5)^2+1) = **0.436**	1/√(|| (1,1)-(1,-0.5) || ^{2}+1) = 1/√((1-1)^2 + (1-(-0.5))^2+1) = **0.555**
2	1	-1	1/√(|| (1,-1)-(-1,0.5) || ^{2}+1) = 1/√((1-(-1))^2 + (-1-0.5)^2+1) = **0.371**	1/√(|| (1,-1)-(1,-0.5) || ^{2}+1) = 1/√((1-1)^2 + (-1-(-0.5))^2+1) = **0.894**
3	0	-1	1/√(|| (0,-1)-(-1,0.5) || ^{2}+1) = 1/√((0-(-1))^2 + (-1-0.5)^2+1) = **0.485**	1/√(|| (0,-1)-(1,-0.5) || ^{2}+1) = 1/√((0-1)^2 + (-1-(-0.5))^2+1) = **0.667**
4	-1	-1	1/√(|| (-1,-1)-(-1,0.5) || ^{2}+1) = 1/√((-1-(-1))^2 + (-1-0.5)^2+1) = **0.555**	1/√(|| (-1,-1)-(1,-0.5) || ^{2}+1) = 1/√((-1-1)^2 + (-1-(-0.5))^2+1) = **0.436**
5	-1	1	1/√(|| (-1,1)-(-1,0.5) || ^{2}+1) = 1/√((-1-(-1))^2 + (-0.5)^2+1) = **0.894**	1/√(|| (-1,1)-(1,-0.5) || ^{2}+1) = 1/√((-1-1)^2 + (1-(-0.5))^2+1) = **0.371**
6	0	1	1/√(|| (0,1)-(-1,0.5) || ^{2}+1) = 1/√((0-(-1))^2 + (1-0.5)^2+1) = **0.667**	1/√(|| (0,1)-(1,-0.5) || ^{2}+1) = 1/√((0-1)^2 + (1-(-0.5))^2+1) = **0.485**



Please use hyperspheric RBF kernel to (c=1.5) convert the six instances into new space. Report the new feature values for each instance.

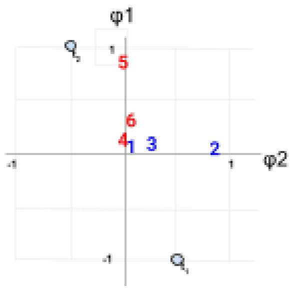
$$\varphi(||x-t||) = \begin{cases} 1 & \text{if } ||x-t|| \leq c \\ 0 & \text{if } ||x-t|| > c \end{cases}$$

	F_1	F_2	φ_1	φ_2
1	1	1	$\|(1,1)-(-1,0.5)\| = \mathrm{sqrt}((1-(-1))^2 + (1-0.5)^2) = \mathbf{2.062 \longrightarrow 0}$	$\|(1,1)-(1,-0.5)\| = \mathrm{sqrt}((1-1)^2 + (1-(-0.5))^2) = \mathbf{1.500 \longrightarrow 1}$
2	1	-1	$\|(1,-1)-(-1,0.5)\|^2+1) = \mathrm{sqrt}((1-(-1))^2 + (-1-0.5)^2) = \mathbf{2.500 \longrightarrow 0}$	$\|(1,-1)-(1,-0.5)\| = \mathrm{sqrt}((1-1)^2 + (-1-(-0.5))^2) = \mathbf{0.500 \longrightarrow 1}$
3	0	-1	$\|(0,-1)-(-1,0.5)\|^2+1) = \mathrm{sqrt}((0-(-1))^2 + (-1-0.5)^2) = \mathbf{1.803 \longrightarrow 0}$	$\|(0,-1)-(1,-0.5)\| = \mathrm{sqrt}((0-1)^2 + (-1-(-0.5))^2) = \mathbf{1.118 \longrightarrow 1}$
4	-1	-1	$\|(-1,-1)-(-1,0.5)\|^2+1) = \mathrm{sqrt}((-1-(-1))^2 + (-1-0.5)^2) = \mathbf{1.500 \longrightarrow 1}$	$\|(-1,-1)-(1,-0.5)\| = \mathrm{sqrt}((-1-1)^2 + (-1-(-0.5))^2) = \mathbf{2.062 \longrightarrow 0}$
5	-1	1	$\|(-1,1)-(-1,0.5)\|^2+1) = \mathrm{sqrt}((-1-(-1))^2 + (-0.5)^2) = \mathbf{0.500 \longrightarrow 1}$	$\|(-1,1)-(1,-0.5)\| = \mathrm{sqrt}((-1-1)^2 + (1-(-0.5))^2) = \mathbf{2.500 \longrightarrow 0}$
6	0	1	$\|(0,1)-(-1,0.5)\|^2+1) = \mathrm{sqrt}((0-(-1))^2 + (1-0.5)^2) = \mathbf{1.118 \longrightarrow 1}$	$\|(0,1)-(1,-0.5)\| = \mathrm{sqrt}((0-1)^2 + (1-(-0.5))^2) = \mathbf{1.803 \longrightarrow 0}$

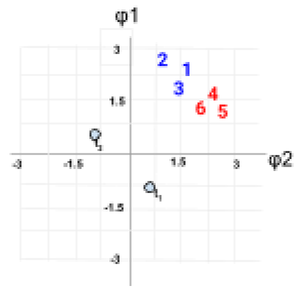


For comparison, the result of each RBF side by side:

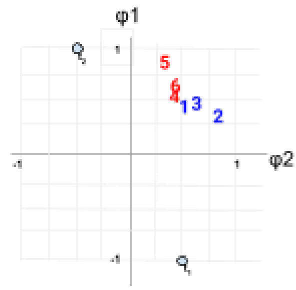
Gaussian



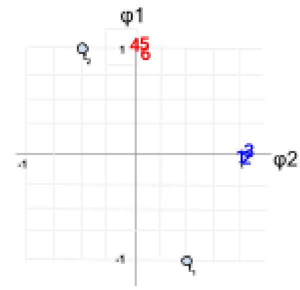
Multiquadratics



Inverse Multiquadratics



Hyperspheric

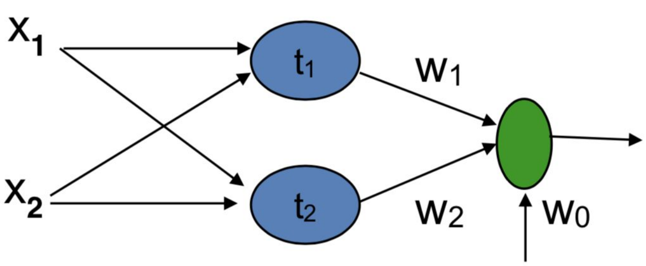


Question 2 [3 pts]

When building an RBF network to solve the XOR problem with input and output given as follows, where X_1 and X_2 are features and Y is the output.

X_1	X_2	Y
0	0	-1
1	0	1
0	1	1
1	1	-1

Assume the RBF network is showing as follows, and we use Gaussian RBF function with σ=1, and t_1=(0.1, 0.1), t_2=(0.9, 0.9). Please use pseudo-inverse to calculate the weight values W=[w_0, w_2, w_3] of the output node [2 pts], and validate your results with respect to the four instances [1 pt]



$$\varphi_1 = exp\left(\frac{-||x-t_1||^2}{\sigma^2}\right) = exp\left(\frac{-||x-t_1||^2}{1^2}\right) = exp(-||x-t_1||^2)$$
$$\varphi_2 = exp\left(\frac{-||x-t_2||^2}{\sigma^2}\right) = exp\left(\frac{-||x-t_2||^2}{1^2}\right) = exp(-||x-t_2||^2)$$

X_1	X_2	φ_1	φ_2
0	0	exp(- || (0,0)-(0.1,0.1) || ^{2}) = exp(-((0-0.1)^2 + (0-0.1)^2)) = **0.980**	exp(- || (0,0)-(0.9,0.9) || ^{2}) = exp(-((0-0.9)^2 + (0-0.9)^2)) = **0.198**
1	0	exp(- || (1,0)-(0.1,0.1) || ^{2}) = exp(-((1-0.1)^2 + (0-0.1)^2)) = **0.440**	exp(- || (1,0)-(0.9,0.9) || ^{2}) = exp(-((1-0.9)^2 + (0-0.9)^2)) = **0.440**
0	1	exp(- || (0,1)-(0.1,0.1) || ^{2}) = exp(-((0-0.1)^2 + (1-0.1)^2)) = **0.440**	exp(- || (0,1)-(0.9,0.9) || ^{2}) = exp(-((0-0.9)^2 + (1-0.9)^2)) = **0.440**
1	1	exp(- || (1,1)-(0.1,0.1) || ^{2}) = exp(-((1-0.1)^2 + (1-0.1)^2)) = **0.198**	exp(- || (1,1)-(0.9,0.9) || ^{2}) = exp(-((1-0.9)^2 + (1-0.9)^2)) = **0.980**

$$\begin{bmatrix} 1 & \varphi_1(||X_1 - t_1||) & \varphi_2(||X_1 - t_2||) \\ 1 & \varphi_1(||X_2 - t_1||) & \varphi_2(||X_2 - t_2||) \\ 1 & \varphi_1(||X_3 - t_1||) & \varphi_2(||X_3 - t_2||) \\ 1 & \varphi_1(||X_4 - t_1||) & \varphi_2(||X_4 - t_2||) \end{bmatrix} \times \begin{bmatrix} w_0 & w_1 & w_2 \end{bmatrix}^T = \begin{bmatrix} Y_1 \\ Y_2 \\ Y_3 \\ Y_4 \end{bmatrix}^T$$

$$\begin{bmatrix} 1 & 0.980 & 0.198 \\ 1 & 0.440 & 0.440 \\ 1 & 0.440 & 0.440 \\ 1 & 0.165 & 0.980 \end{bmatrix} \times \begin{bmatrix} w_0 & w_1 & w_2 \end{bmatrix}^T = \begin{bmatrix} -1 \\ 1 \\ 1 \\ -1 \end{bmatrix}^T$$

$$\begin{bmatrix} w_0 & w_1 & w_2 \end{bmatrix}^T = \Phi^+ \times \begin{bmatrix} -1 \\ 1 \\ 1 \\ -1 \end{bmatrix}^T$$

$$\begin{bmatrix} w_0 & w_1 & w_2 \end{bmatrix}^T = \begin{bmatrix} -1.477 & 1.977 & 1.977 & -1.477 \\ 2.317 & -1.678 & -1.678 & 1.038 \\ 1.038 & -1.678 & -1.678 & 2.317 \end{bmatrix} \times \begin{bmatrix} -1 \\ 1 \\ 1 \\ -1 \end{bmatrix}^T$$

$$\begin{bmatrix} w_0 & w_1 & w_2 \end{bmatrix}^T = \begin{bmatrix} 6.908 \\ -6.711 \\ -6.711 \end{bmatrix}$$

Validating the instances:

X_1	X_2	$\sum w_i \varphi_i$	Predicted Y
0	0	1 x (6.908) + 0.980 x (-6.711) + 0.198 x (-6.711) = -0.998 < 0	**-1**
1	0	1 x (6.908) + 0.440 x (-6.711) + 0.440 x (-6.711) = 0.996 > 0	**1**
0	1	1 x (6.908) + 0.440 x (-6.711) + 0.440 x (-6.711) = 0.996 > 0	**1**
1	1	1 x (6.908) + 0.198 x (-6.711) + 0.980 x (-6.711) = -0.998 < 0	**-1**

Question 3 [3 pts]

Table 2 shows a 5x5 synthetic image, and Table 3 shows a 2x2 filter.

Table 2:

4	3	4	0	4
3	5	5	1	6
2	7	3	1	5
6	5	4	0	6
6	3	5	0	4

Table 3

1	-1
1	-1

Please apply the filter in Table 3 to the image in Table 2 (using convolutional filter and stride=1), and show the output of the resulting image.

4-3+3-5=-1	3-4+5-5=-1	4-0+5-1=8	0-4+1-6=-9
3-5+2-7=-7	5-5+7-3=4	5-1+3-1=6	1-6+1-5=-9
2-7+6-5=-4	7-3+5-4=5	3-1+4-0=6	1-5+0-6=-10
6-5+6-3=4	5-4+3-5=-1	4-0+5-0=9	0-6+0-4=-10

-1	-1	8	-9
-7	4	6	-9
-4	5	6	-10
4	-1	9	-10

Please apply 2x2 max pooling to the above result, and report the resulting image.

Note: using stride = 2, as we did in the class example.

-1	-1	8	-9
-7	4	6	-9
-4	5	6	-10
4	-1	9	-10

4	8
5	9

Assume a flattening procedure is applied to the above result, please report the final flattened features from the above process (i.e., after the max pooling).

4	8
5	9

4
8
5
9

Explain what is the possible role of the filter in Table 3, and how does the filter achieve the goal through the convolution process.

The filter is detecting vertical lines in the image. It achieves that by convolving the image with the filter values. Since the filter has "1" in the first vertical column, the convolution operation will result in higher values for sequences of vertical pixels in the image.

This is illustrated below with the simplified image of the letter T written in black ink, where 1 represents the pixels that have content (are black) and 0 represents the empty pixels. The convolution operation operation results in high values for the vertical part of the letter T.

Image of letter T

1	**1**	**1**	**1**	**1**
0	0	**1**	0	0
0	0	**1**	0	0
0	0	**1**	0	0
0	0	**1**	0	0

Filter

1	-1
1	-1

Convolution

0	-1	**1**	0
0	-2	**2**	0
0	-2	**2**	0
0	-2	**2**	0

Question 4 [2 pts]

Please summarize the major differences between a convolutional neural network (CNN) vs. a fully connected multi-layer neural network in terms of their structures, learning efficiency, and functionalities.

Structural differences

- Fully connected network: every single hidden node sees every single input.
- Convolutional network: a node sees only the input values selected by the filter.

Learning efficiency

- Fully connected network: each node has its own weight that has to be trained.
- Convolutional network: weight values are shared, resulting in fewer parameters to train, thus faster training.

Functionality

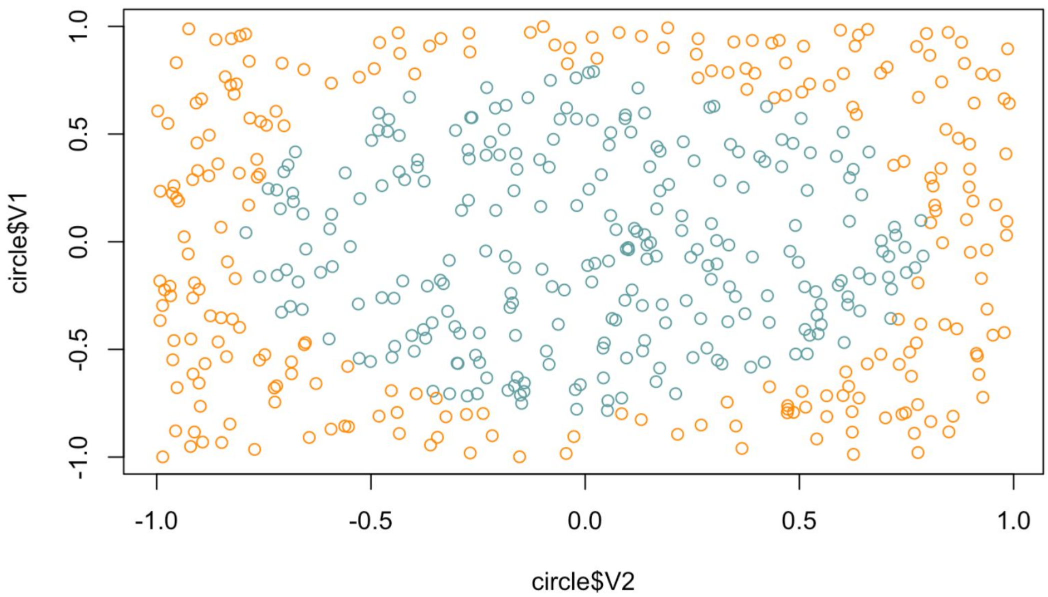
- Fully connected network: layers don't have a particular function; the layers of the network cannot be reused for other applications.
- Convolutional network: layers are responsible feature extraction, recognizing patterns. With these specialized layers, convolutional networks become invariant to shift and distortion. The lower layers of the network can be reused for multiple applications that need feature (pattern) extraction, for example different types of image classification.

Question 5 [5 pts]

The MLBENCH (machine learning benchmark problem) is an R package which can generate synthetic data for test. Please revise the R node in the MLBENCH: ML Benchmark data generator R Notebook [hml] and RBF Network training & classification R Notebook [html] to implement following tasks

Please find the code for this question in cap6619-homework3-question5.Rmd and cap6619-homework3-question5.nb.html.

Please use MLBENCH to generate a two-class “circle” dataset (with 500 instances). Please show all 500 instances in one plot and color instances according to the class label they belonging to. [1 pt].



Please split the generated “circle” dataset into 30% training vs. 70% test datasets. Train an RBF netowrk (using Gaussian RBF kernel and σ=1), and validate the classification accuracy on the test dataset. Please report the classification results (including confusion matrix and accuracy) with respect to 2 centers, 5 centers, and 10 centers [1 pts].

Number of kernels	Confusion matrix	Accuracy

2	`predictions`
	`test_set_labels  -1    1`
	`-1 167   16`
	`1   22  145`
		0.8914

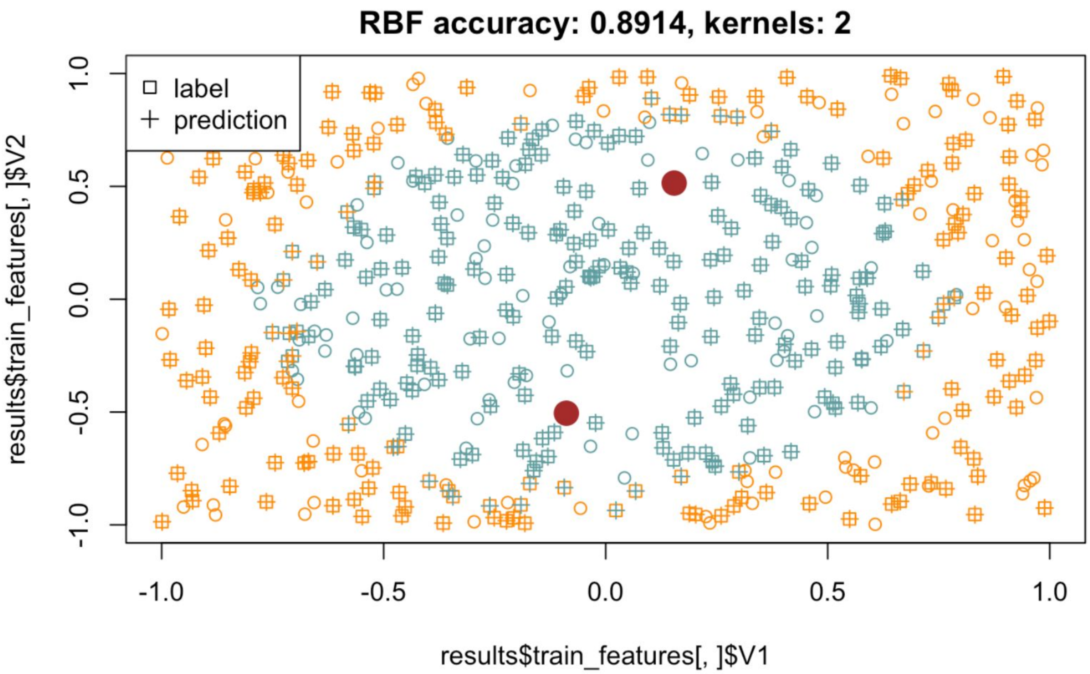
5	`predictions`
	`test_set_labels  -1    1`
	`-1 175    8`
	`1   12  155`
		0.9429

10	`predictions`
	`test_set_labels  -1    1`
	`-1 173   10`
	`1   12  155`
		0.9371

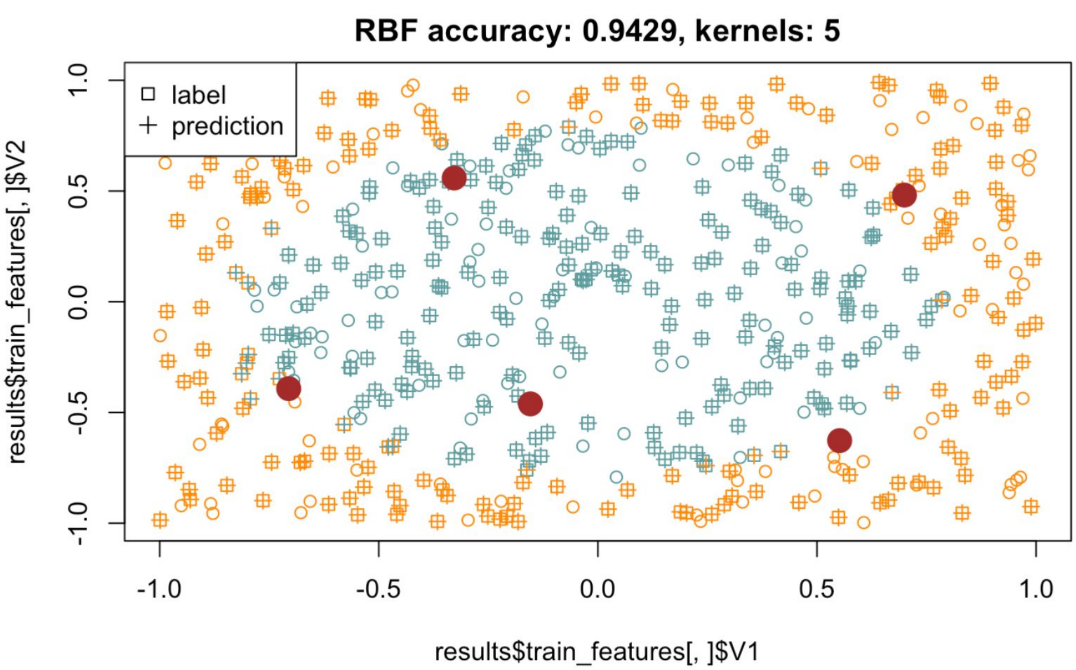
Please also color-code and show all training and test instances in the same plot, and also mark RBF centers and misclassified instances [1pt].

- Correctly classified instances: 
- Incorrectly classified instances: 

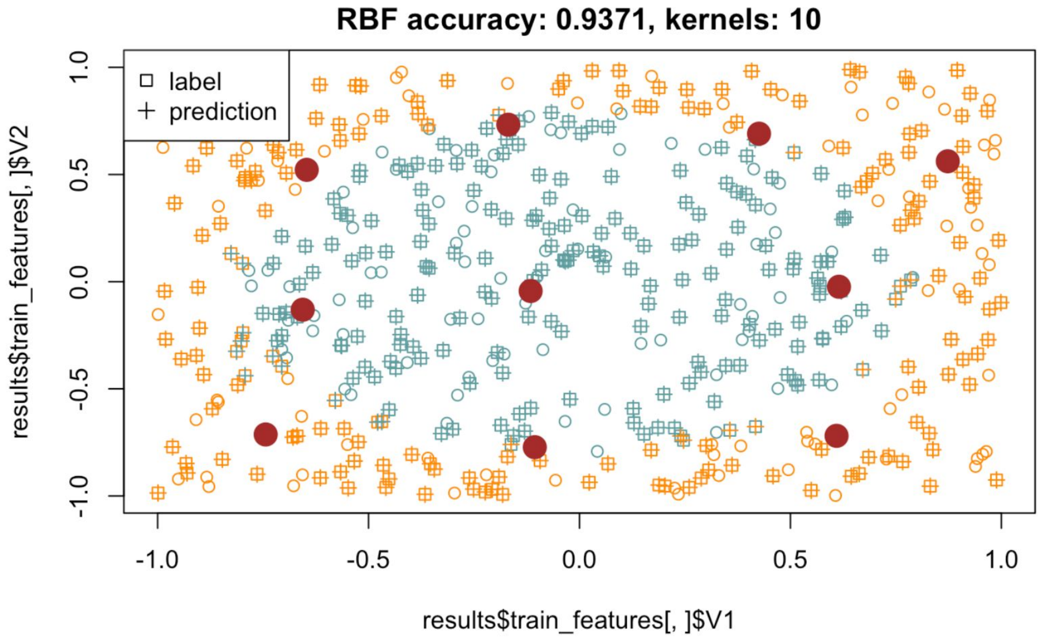
2 kernels



5 kernels



10 kernels



For the same “circle” dataset (with 30% training vs. 70% test), please train a two hidden layer neural networks (with 5 hidden nodes for each layer) to validate its classification accuracy on the 70% test instances (report confusion matrix and accuracy) [1 pt].

```
         actual
predicted  -1   1
       -1 181   5
       1    2 162
```

Accuracy: 0.98

Compare multi-layer neural network results vs. RBF network (10 centers), and explain why one method outperforms the other method [1 pt]

	Confusion matrix (test data)	Accuracy (test data)	Accuracy (training data)

RBF with ten kernels	`predictions`
	`test_set_labels  -1   1`
	`-1 173  10`
	`1   12 155`
		0.937	0.96

Neural network with two layers, five neurons each	`actual`
	`predicted  -1   1`
	`-1 181   5`
	`1    2 162`
		0.980	1.00

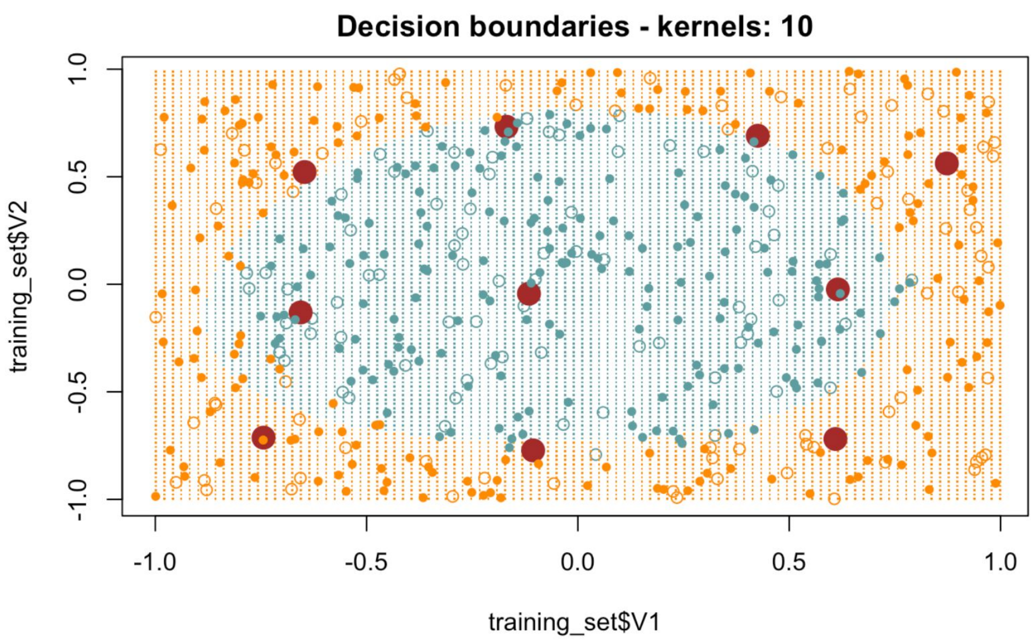
The multi-layer network outperforms the RBF network with ten centers because it is has higher accuracy on the test data and overfits slightly less. Although both models show sign of overtiffting, the multi-layer network seems to overfit less (the difference between the training set accuracy and test set accuracy is smaller).

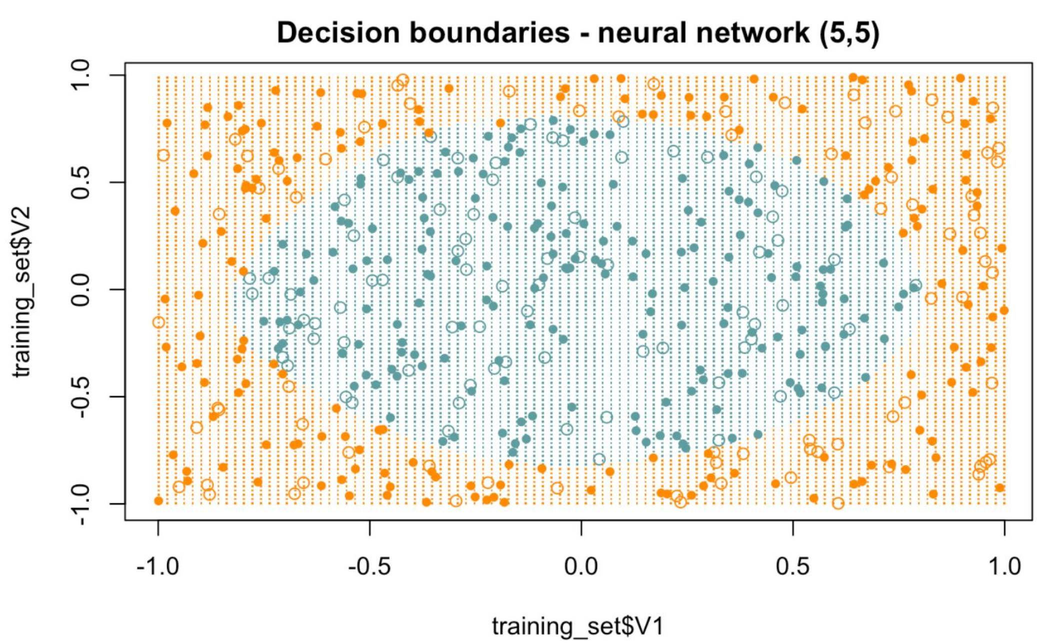
To help visualize that we built decision boundaries for both cases, shown below. The decision boundaries for the RBF network with ten centers is shown first, followed by the decision boundaries for the neural network.

Training and test data is shown in both graphs:

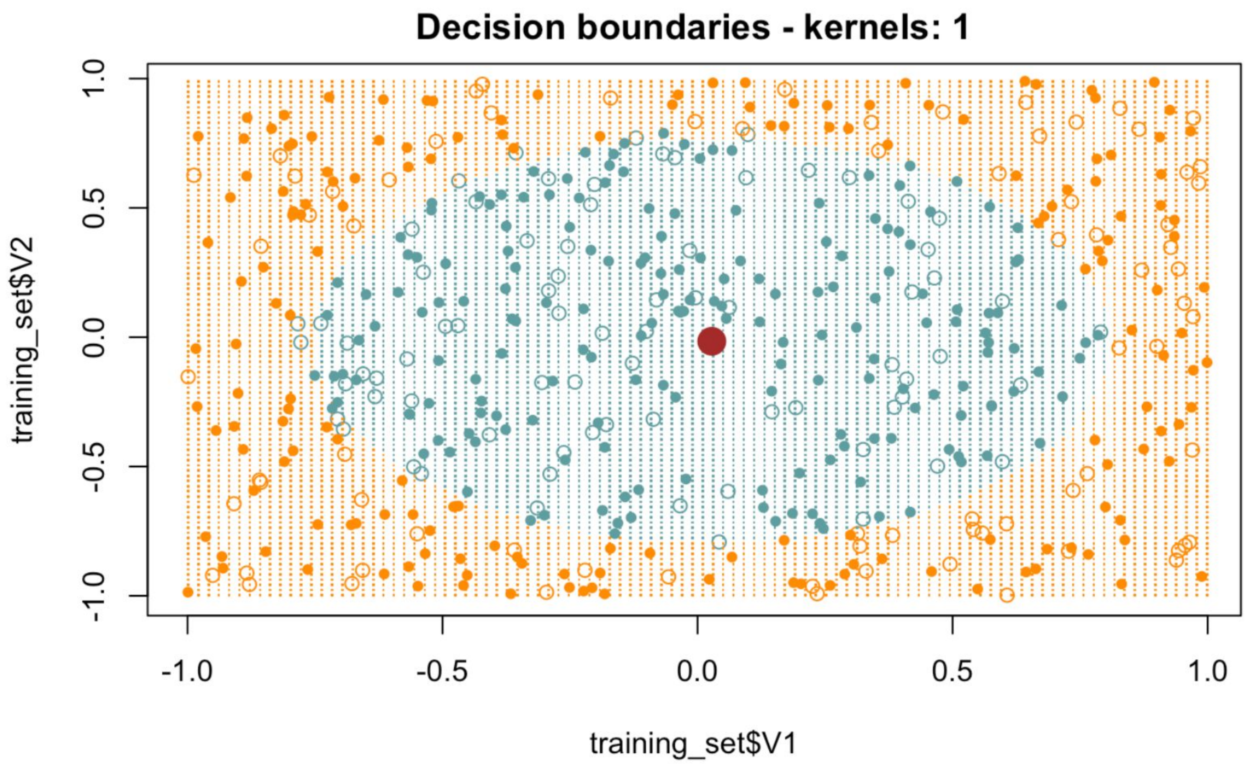
- Empty circle: training data
- Solid circle: test data

Comparing the pictures we can see that the neural network's decision boundaries capture the "circleness" of the data better (its decision boundaries fit the data better). The decision boundaries for the RBF with ten centers shows a more elliptical shape that doesn't match the underlying representation of the data (a circle) as well as the neural network does. The result is a higher probability of misclassifying test data points because of overfitting.



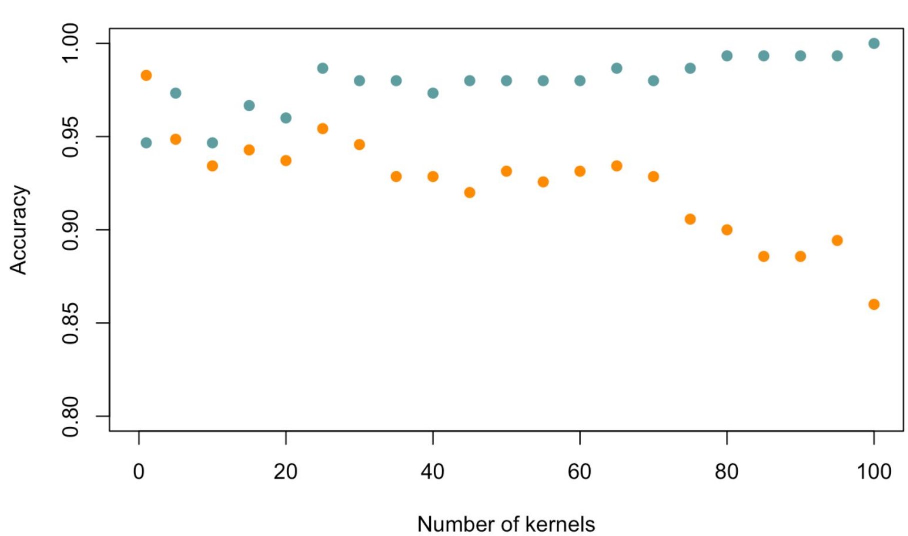


For this data set, the "circle in a square", the ideal case for an RBF network is one center at coordinate (0,0). We can see that in the RBF network case with one center. In the picture below we have the decision boundary for an RBF with one center. It shows how this network configuration captures the underlying structure of the data (a circle).

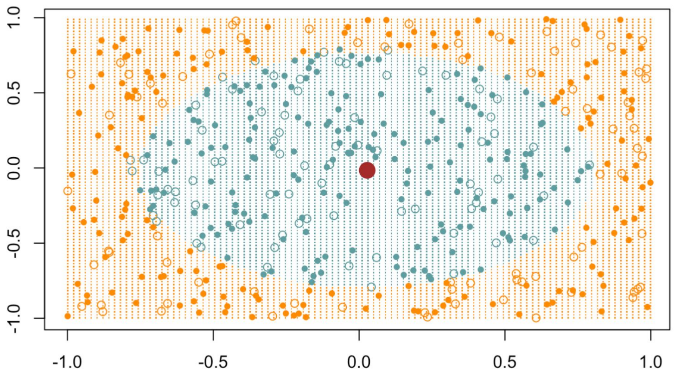
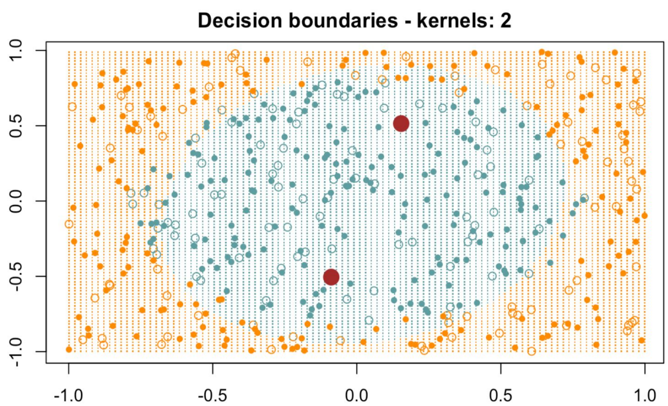
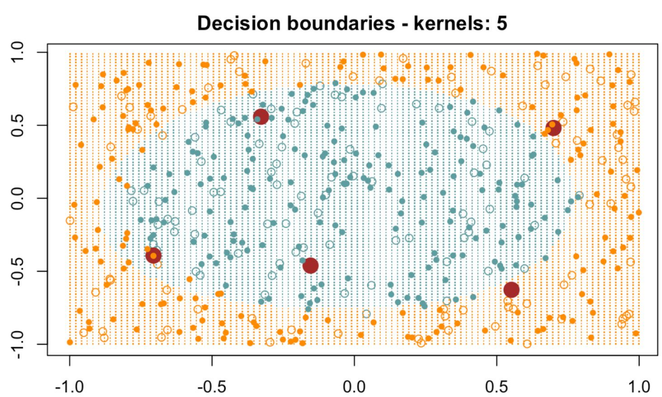
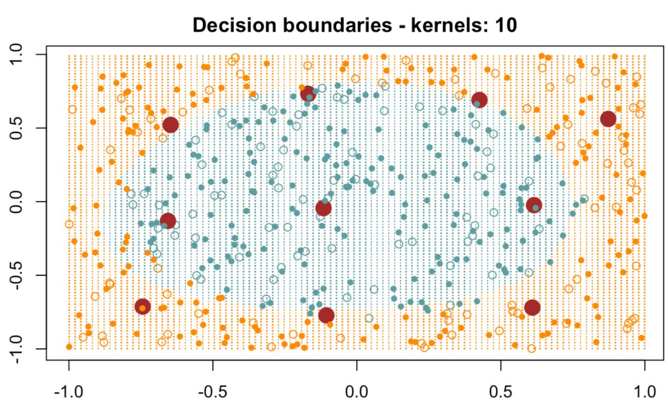


This RBF network configuration (one kernel) results in an accuracy of 0.9829, higher than any other configuration.

Given the underlying structure of the data (a circle), adding more centers to the network reduces its accuracy on the test data, as the network overfits more and more. This can be seen in the graphs below, showing the accuracy of the RBF network vs. number of centers. The blue dots show training accuracy and the orange dots show test accuracy. The lines diverge more and more as we add kernels, with accuracy trending down (a sign of overfitting).



We can visualize that effect in the decision boundaries. As we add centers, we overfit more and more.

Kernels	**Decision boundaries**
1	
2	
5	
10	

The effect of randomness in this analysis

Although the results shown above for question 5 result in the conclusion that the neural network is better, they also show that it's only marginally better.

Given how randomness can affect the final results of a neural network (for example, how weight values are randomly set for the first round of training), I also tried a few other scenarios to see how the networks behave.

The notebook has `set.seed()` at the start to make the results repeatable across runs and environments (recommended practice to make research reproducible - see for example [this source]() and [this other source]()).

Changing the seed value also changes the results of the experiments, as expected. However, because in this case the results are so close, it can also end up flipping the conclusion (sometimes the RBF network does better, sometimes the neural network does better).

Given the small training and test set we have and the fact that we are not using more sophisticated mechanisms to choose a network (e.g. cross-validation with folding), the only conclusion we can reach from the experiments is that the networks perform roughly the same for a data set with this distribution.

As an experiment to check those assumptions, the same tests were executed with `set.seed(1234)`. The table below shows the results, comparing with the results from the homework (using `set.seed(123)`). The table shows that the neural network performs worse this time around, while the RBF performs better.

Accuracy

	RBF 10 kernels		**Neural network**	
	Test data	Training data	Test data	Training data
set.seed(123) - homework	0.94	0.96	0.98	1.00
set.seed(1234) - this test	0.97	0.96	0.93	1.00

This is also visible in the new decision boundaries for the neural network. It shows the boundaries going significantly more outside of the underlying distribution of the data (the circle).