Part 1

1. Supervised, Unsupervised and Reinforcement Learning

2. By using a “bias weight” which adds an offset to the weighted sum.

3. $k$ is the number of stored data vectors that vote for the decision.

4. The first principal component is in the direction of maximum variance and the second component is orthogonal to the first one.

5. $10 \times 10$.

6. The solid curve shows the training error and the dashed curve shows the generalization error.

7. 
   \[ \Delta w(t) = \alpha \Delta w(t - 1) - \eta \frac{\partial \epsilon(t)}{\partial w} \]

8. In order to have the possibility to find an even better policy. The Bias-variance dilemma)

9. Reproduction, crossover, and mutation

10. Axon
11. Given the signals 1, $x$, and $x^2$, we can create an arbitrary second order polynomial. The decision boundaries are shown with vertical lines, where the output function crosses zero.

12. $\varphi(x)^T \varphi(y) = x_1^2y_1^2 + x_2^2y_2^2 + 2x_1x_2y_1y_2 = (x_1y_1 + x_2y_2)^2 = (x^T y)^2$ (1)

13. Maximum correlation is obtained for a projection orthogonal to the noise:

14. This is given by the eigen value problem $Ax = x$ with the condition $\sum_i x_i = 1$, where $A$ is the state transition probability matrix. This gives $p(1) = x_1 = 0.25$. 
15. The optimal hyper plane is the line that best separates the two classes, i.e. the line that is as far away from the closest points as possible. The support vectors are the samples that are closest to this plane.

Part 3

16. a) Set

\[ \mathbf{m} = \begin{pmatrix} m_1 \\ m_2 \\ m_3 \end{pmatrix} \]

If we use \( \mathbf{m}(t) \) and \( \mathbf{m}(t-1) \) as input data, CCA will find the three linear combinations of \( m_1, m_2 \) and \( m_3 \) having the largest autocorrelation at one sample distance. The resulting output data is the coefficients for these linear combinations (in two almost identical sets; one for \( \mathbf{m}(t) \) and one for \( \mathbf{m}(t-1) \)) and the corresponding correlations. The linear combinations are the eigenvectors and the correlations are the square root of the eigenvalues of the eigenvalue problem the solution is based upon.

\( \mathbf{m}(t) \) and \( \mathbf{m}(t-1) \) are matrices of the size \( N \times M \), where \( N \) is the number of mixes (3) and \( M \) is the number of samples. The eigenvectors have the size \( N \times 1 \), and there are \( N \) such eigenvectors for \( \mathbf{m}(t) \) and \( N \) for \( \mathbf{m}(t-1) \). You get an eigenvalue (\( \Rightarrow \) a correlation) for each pair of eigenvectors, i.e. \( N \) eigenvalues.

b) If the eigenvectors lie in the columns of \( \mathbf{W}_1 \) and \( \mathbf{W}_2 \), we can write \( \mathbf{s} \approx \mathbf{W}_1^T \mathbf{m} \approx \mathbf{W}_2^T \mathbf{m} \). The components of the eigenvectors are consequently used as coefficients for linear combinations of the mixed signals.

c) If the signals have different time displacements relative to the sound sources, the assumptions for this algorithm to work do not apply. If there is additional noise after the microphone recordings, this will also disturb the separation.
17. (One way to do this is presented in the lectures. One alternative is given below.)

The first principal direction \( \mathbf{w} = (w_1, \ldots, w_n)^T \) is the direction that maximizes the variance \( V(\mathbf{w}^T \mathbf{x}) \). If we first remove the average from each variable in \( \mathbf{x} \), then the variance is given by

\[
V(\mathbf{w}^T \mathbf{x}) = E\left( (\mathbf{w}^T \mathbf{x})^2 \right) = E\left( (\mathbf{w}^T \mathbf{x}) (\mathbf{x}^T \mathbf{w}) \right) = \mathbf{w}^T E(\mathbf{x} \mathbf{x}^T) \mathbf{w} = \mathbf{w}^T \mathbf{C} \mathbf{w}
\]

A simple way to maximize \( \mathbf{w}^T \mathbf{C} \mathbf{w} \) is to scale \( \mathbf{w} \). We therefore have to limit the length of \( \mathbf{w} \) so that only the direction is considered. The problem then becomes

\[
\max \mathbf{w}^T \mathbf{C} \mathbf{w}
\]

with constraint \( \|\mathbf{w}\|^2 = 1 \)

We include the constraint into the target function using a Lagrange-parameter \( \lambda \):

\[
\max \mathbf{w}^T \mathbf{C} \mathbf{w} + \lambda (\|\mathbf{w}\|^2 - 1)
\]

The derivative of the expression above with respect to \( \mathbf{w} \) is

\[
2\mathbf{C} \mathbf{w} + 2\lambda \mathbf{w},
\]

and if we set it to 0 we get the equation

\[
\mathbf{C} \mathbf{w} = \lambda \mathbf{w}
\]

The \( \mathbf{w} \) we want to find is therefore an eigenvector to the covariance matrix \( \mathbf{C} \). If we denote the eigenvectors of \( \mathbf{C} \) as \( \hat{\mathbf{w}}_1, \ldots, \hat{\mathbf{w}}_n \) and use the eigenvector \( \hat{\mathbf{w}}_i \) we get the variance

\[
\hat{\mathbf{w}}_i^T \mathbf{C} \hat{\mathbf{w}}_i = \lambda_i \hat{\mathbf{w}}_i^T \hat{\mathbf{w}}_i = \lambda_i,
\]

i.e. the eigenvalues give the variance in the direction of the corresponding eigenvector. The solution is consequently given by the eigenvector that corresponds to the largest eigenvalue.
18. Exercise A)

Since we want to maximize the reward over time, an appropriate definition for an optimal Q-function is given by the expression

\[ Q^*(x_t, a_t) = r(x_t, a_t) + \gamma V^*(x_{t+1}) \]
\[ = r(x_t, a_t) + \gamma \max_b Q^*(x_{t+1}, b) \]

We can find a solution by traversing the state graph recursively starting from the end.

According to the state graph \( V^*(2) = V^*(4) = V^*(5) = 0 \).

\[ Q^*(1, \text{left}) = 3 + \gamma V^*(2) = 3 \]
\[ Q^*(3, \text{left}) = -10 + \gamma V^*(4) = -10 \]
\[ Q^*(3, \text{right}) = 4 + \gamma V^*(5) = 4 \]
\[ V^*(3) = \max(Q^*(3, \text{left}), Q^*(3, \text{right})) = 4 \]
\[ Q^*(1, \text{right}) = 1 + \gamma V^*(3) = 1 + 4\gamma \]
\[ V^*(1) = \max(Q^*(1, \text{left}), Q^*(1, \text{right})) = \]
\[ \max(3, 1 + 4\gamma) = \begin{cases} 3 & \text{om } \gamma \leq \frac{1}{2} \\ 1 + 4\gamma & \text{om } \gamma > \frac{1}{2} \end{cases} \]

Exercise B)

In this exercise you should use the \( \varepsilon \)-greedy policy and therefore we have to include an expectation value in the definition of the \( Q \)- and \( V \)-function.

\[ Q^\varepsilon(x_t, a_t) = r(x_t, a_t) + \gamma V^\varepsilon(x_{t+1}) \]
\[ = r(x_t, a_t) + \gamma E\{Q^\varepsilon(x_{t+1}, \mu^\varepsilon(x_{t+1}))\} \]

According to the state graph \( V^*(2) = V^*(4) = V^*(5) = 0 \).

\[ Q^*(1, \text{left}) = 3 + \gamma V^*(2) = 3 \]
\[ Q^*(3, \text{left}) = -10 + \gamma V^*(4) = -10 \]
\[ Q^*(3, \text{right}) = 4 + \gamma V^*(5) = 4 \]
\[ V^*(3) = E\{Q(3, \mu^\varepsilon(3))\} = (1 - \varepsilon)4 + \varepsilon(-10) = 4 - 14\varepsilon \]
\[ Q^*(1, \text{right}) = 1 + \gamma V^*(3) = 1 + (4 - 14\varepsilon)\gamma \]
\[ V^*(1) = E\{Q(1, \mu^\varepsilon(1))\} = \]
\[ \begin{cases} (1 - \varepsilon)3 + \varepsilon(1 + \gamma(4 - 14\varepsilon)) & \text{if } 3 > 1 + \gamma(4 - 14\varepsilon) \\ (1 - \varepsilon)(1 + \gamma(4 - 14\varepsilon)) + \varepsilon3 & \text{if } 3 \leq 1 + \gamma(4 - 14\varepsilon) \end{cases} \]

Set \( \varepsilon = \frac{1}{14} \)

\[ V^\varepsilon(1) = \begin{cases} \frac{13}{14}3 + \frac{1}{14}(1 + 3\gamma) & \text{if } 3 > 1 + 3\gamma \\ \frac{13}{14}(1 + 3\gamma) + \frac{3}{14} & \text{if } 3 \leq 1 + 3\gamma \end{cases} \]
a) The update should be as

\[ \Delta w_{ji} = -\eta \frac{\partial \varepsilon}{\partial w_{ji}} \]

where \( \varepsilon \) is defined as

\[ \varepsilon = \frac{1}{2} \sum_k e_k^2 = \frac{1}{2} \sum_k (d_k - y_k)^2 \]

where \( k \) is the number of output neurons.

For the output layer we have

\[ \frac{\partial \varepsilon}{\partial w_{ji}} = \frac{\partial \varepsilon}{\partial e_j} \frac{\partial e_j}{\partial y_j} \frac{\partial y_j}{\partial w_{ji}} = -e_j v_i \]

and for the hidden layer we have

\[ \frac{\partial \varepsilon}{\partial u_{ji}} = \sum_k \frac{\partial \varepsilon}{\partial e_k} \frac{\partial e_k}{\partial y_k} \frac{\partial y_k}{\partial v_j} \frac{\partial v_j}{\partial s_j} \frac{\partial s_j}{\partial u_{ji}} = -\sum_k e_k w_{kj} \sigma'(s_j) x_i \]

So the update of the output layer becomes:

\[ \Delta w_{ji} = \eta e_j v_i \]

and for the hidden layer:

\[ \Delta u_{ji} = \eta \sum_k e_k w_{kj} \sigma'(s_j) x_i \]

b) Without the activation function the two layers could be combined to a single layer. In practice we would have a single layer network and therefore only capable of linear decision boundaries.

c) While training, the output data are calculated for a set of training samples with known correct classifications. The output of the network is compared to the known correct output and the weights of the network are updated according to the expressions above. While classifying, the output data are calculated for one or several unknown samples. The class of the unknown sample is decided by the output neuron emitting the largest value.