Martin Lundfall, Henri Bunting, Malte Siemers, Patrik Bey

Exercise sheet 08 - Machine Intelligence I

8.1

Given:

$$C_{(p,N)} = 2 * \sum_{k=0}^{N-1} \binom{p-1}{k} \tag{1}$$

Prove:

$$C_{(p,N)} + C_{(p,N-1)} = C_{(p+1,N)} \tag{2}$$

Proof:
By the definition of C, we can rewrite into sums:

$$2 * \sum_{k=0}^{N-1} \binom{p-1}{k} + 2 * \sum_{k=0}^{N-2} \binom{p-1}{k} = 2 * \sum_{k=0}^{N-1} \binom{p}{k} \tag{3}$$

We can then divide out the 2:

$$\sum_{k=0}^{N-1} \binom{p-1}{k} + \sum_{k=0}^{N-2} \binom{p-1}{k} = \sum_{k=0}^{N-1} \binom{p}{k} \tag{4}$$

We know:

$$\sum_{k=0}^{0} \binom{p-1}{k} = \binom{p-1}{0} = 1 \tag{5}$$

So we pull out the first k=0 terms:

$$1 + \sum_{k=1}^{N-1} \binom{p-1}{k} + \sum_{k=0}^{N-2} \binom{p-1}{k} = 1 + \sum_{k=1}^{N-1} \binom{p}{k} \tag{6}$$

But then we subtract them to get rid of them and rewrite the second term so that k starts at 1:

$$\sum_{k=1}^{N-1} \binom{p-1}{k} + \sum_{k=1}^{N-1} \binom{p-1}{k-1} = \sum_{k=1}^{N-1} \binom{p}{k} \tag{7}$$

Since the sums have the same indices, we can reform them into a single sum:

$$\sum_{k=1}^{N-1} (\binom{p-1}{k} + \binom{p-1}{k-1}) = \sum_{k=1}^{N-1} \binom{p}{k} \tag{8}$$

And after dropping the sums we get:

$$\binom{p-1}{k} + \binom{p-1}{k-1} = \binom{p}{k} \tag{9}$$

Which is the same as the binomial theorem:

$$\binom{r}{q} + \binom{r}{q-1} = \binom{r+1}{q} \tag{10}$$

8.2

a)

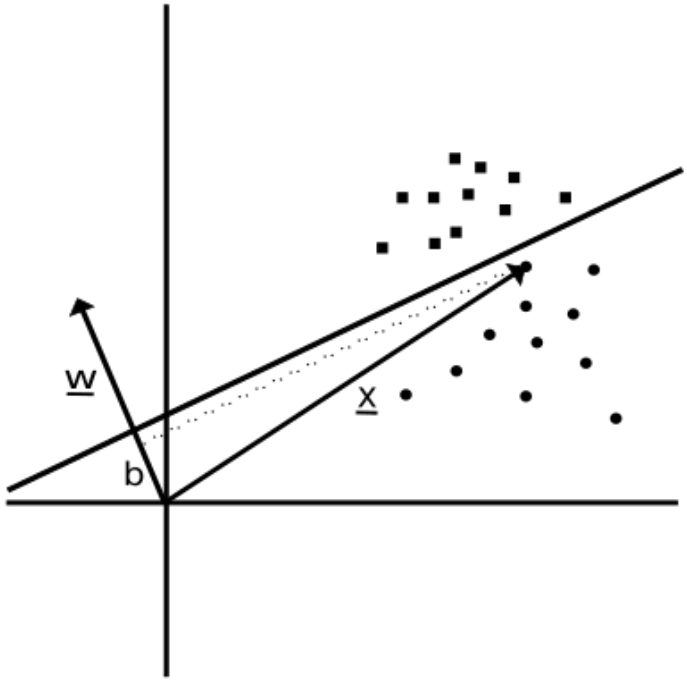

Figure 1: Geometry of a linear classifier. $\underline{\mathbf{x}}$ is the vector to the location of a data point, $\underline{\mathbf{w}}$ the weight vector of the neuron and **b** the distance from the origin to the classification boundary.

The geometry of a binary classifier of form $y = sign(\underline{\mathbf{w}}^T\underline{\mathbf{x}} - b)$ is illustrated above. The decision boundary is orthogonal to the weight vector $\underline{\mathbf{w}}$ with distance b to the origin. $\underline{\mathbf{w}}^T\underline{\mathbf{x}}$ is the projection of $\underline{\mathbf{x}}$ on $\underline{\mathbf{w}}$ and b shifts the decision boundary according to the classification problem, so that $sign(\underline{\mathbf{w}}^T\underline{\mathbf{x}} - b)$ becomes negative for points on one side of the boundary and positive for the other.

b)

8.3

a)

A margin can be thought of as the 'width' of the decision boundary of the linear connectionist neuron, or how close we allow the boundary line to come to the data points. The effect of increasing the margin is that the number of possible classifying lines decreases which effectively reduces the complexity of the classifier. If this can be done without affecting the training/empirical error of the classifier, it can reduce the generalization error.

b)

The Euclidean distance $d(x^\alpha, w, b)$ from a sample $x^{(\alpha)}$ to the decision boundary line $L = \{x \in X | w^T x + b = 0\}$ is given by

$$d(x^\alpha, w, b) = |\frac{w^T x^{(\alpha)}}{||w||} + \frac{-b}{||w||}| = \frac{1}{||w||}\left(w^T x^{(\alpha)} + b\right) \tag{11}$$

With the constraint that $w^T x^{(\alpha)} + b \leq 1$ for all $x \in X$, we get

$$d(x^\alpha, w, b) \leq \frac{1}{||w||} \tag{12}$$

c)

The *primal optimization problem* offers a method of maximizing the margin of the classifier while keeping the constraint that we still classify all training points correctly. This is done by using the KuhnTucker conditions, a generalization of Lagrange multipliers, which involves minimizing the Lagrange equation:

$$L_{(x_k, \{\lambda_k\})} = f_0(x) + \sum_{k=1}^{m} \lambda_k f_k(x) \tag{13}$$

Where f_k are constraints expressed in the form that they must be smaller than or equal to zero, and $f_0(x)$ is the function that we want to minimize. In our case, the constraint set by normalizing the margin gives $f(x) = -(y_T(w^T x + b) - 1)$ on this form. Since we want to maximize the margin, we want to minimize $f_0(x) = \frac{||w||^2}{2}$.
The aim is to minimize the Lagrangian function, and we do this by first minimizing with respect to w. This will result us an expression on the form:

$$w = \sum_{\alpha=1}^{p} \lambda_\alpha y_T^{(\alpha)} x^{(\alpha)} \tag{14}$$

Where α is indexing over p datapoints. Then, minimizing with respect to b we get the following constraint on λ and y_T:

$$0 = -\sum_{\alpha=1}^{p} \lambda_\alpha y_T^{(\alpha)} \tag{15}$$

Putting these results back into the Lagrange equation, we attain:

$$L_{(x_k,\{\lambda_k\})} = -\frac{1}{2}\sum_{\alpha,\beta=1}^{p} \lambda_\alpha \lambda_\beta y_T^{(\alpha)} y_T^{(\beta)} (x^{(\alpha)})^T x^{(\beta)} + \sum_{\alpha=1}^{p} \lambda_\alpha \tag{16}$$

The Khuhn-Tucker conditions implies that w can be given as a linear combination of the data points:

$$w = \sum_{\alpha}^{p} \lambda_\alpha y_T^{(\alpha)} x^{(\alpha)} \tag{17}$$

and the b can be given by looking at the support vectors, i. e. the points that lie closest to the decision boundary and therefore satisfy:

$$y_T^{(\alpha)}(w^T x^{(\alpha)} + b) = 1 \tag{18}$$

And we get:

$$b = \frac{1}{N_{SV}} \sum_{i=1}^{N_{SV}} (w^T x_i - y_i) \tag{19}$$

Where i is indexing over the number of support vectors, N_{SV}.

8.4

0.1 a)

Since we have that the support vectors, i.e. the datapoints lying closest to the decision boundary satisfy $y_T^{(\alpha)}(w^T x^{(\alpha)} + b) = 1$, the remaining datapoints will require λ to be zero in order to minimize the lagrangian. Therefore, it turns out that these are the only vectors we really have to pay attention to, and the others are already 'decided' by classifying with respect to the support vectors.

0.2 b)

In Exercise 8.3, we derived the following conditions:

$$w = \sum_{\alpha=1}^{p} \lambda_\alpha y_T^{(\alpha)} x^{(\alpha)} \tag{20}$$

$$0 = -\sum_{\alpha=1}^{p} \lambda_\alpha y_T^{(\alpha)} \tag{21}$$

We put this into the Lagrangian:

$$\begin{aligned} L_{(x_k,\{\lambda_k\})} = \frac{1}{2} \sum_{\alpha,\beta=1}^{p} \lambda_\alpha \lambda_\beta y_T^{(\alpha)} y_T^{(\beta)} (x^{(\alpha)})^T x^{(\beta)} - \\ \sum_{\alpha=1}^{p} \lambda_\alpha y_T^{(\alpha)} \left(\sum_{\beta=1}^{p} \lambda_\beta y_T^{(\beta)} (x^{(\beta)})^T \right) x^{(\alpha)} - \sum_{\alpha=1}^{p} \lambda_\alpha y_T^{(\alpha)} - \sum_{\alpha=1}^{p} \lambda_\alpha \end{aligned} \tag{22}$$

The penultimate term term will disappear thanks to the constraint given by optimizing wwith respect to b, and we see that the first two terms are the same except for the constant, and we arrive at the given expression.

8.5