Solutions Bonus Exercise

Question 1

The first two samples given in the notebook (alphabet and simple sentence) can be solved with a single LSTM layer. Adding nodes makes it better. Of course adding enough nodes it may start to trivially memorise the input.

```
batch_input_shape=(batch_size, X.shape[1], X.shape[2])
model.add(LSTM(30, stateful=True, batch_input_shape=batch_input_shape))
model.add(Dense(y.shape[1], activation='softmax'))
```

The full notebook solution can be found in Moodle.

Question 2

The loss function is given by

$$L(\theta_1, \theta_2) = \frac{(\theta_1 - 2)^2}{4} + (\theta_2 - 2)^2 + \lambda \mid \theta \mid$$

solving for optimal θ_1 using subgradients and also the fact that $\lambda > 0$, we get

$$\frac{\partial L(\theta_1, \theta_2)}{\partial \theta_1} = \frac{\theta_1}{2} - 1 + \begin{cases} -\lambda & \text{if } \theta_1 < 0 \\ [-\lambda, \lambda] & \text{if } \theta_1 = 0 \\ \lambda & \text{if } \theta_1 > 0 \end{cases}$$

$$\frac{\partial L(\theta_1, \theta_2)}{\partial \theta_1} = \begin{cases} \frac{\theta_1}{2} - 1 - \lambda & \text{if } \theta_1 < 0 \\ [-1 - \lambda, -1 + \lambda] & \text{if } \theta_1 = 0 \\ \frac{\theta_1}{2} - 1 + \lambda & \text{if } \theta_1 > 0 \end{cases}$$

$$\theta_1 = \begin{cases} 0 & \text{if } \lambda > 1 \\ 2(1 - \lambda) & \text{if } 0 < \lambda < 1 \end{cases}$$

Similarly, we solve for θ_2 as follows:

$$\frac{\partial L(\theta_1, \theta_2)}{\partial \theta_2} = \begin{cases} 2\theta_2 - 4 - \lambda & \text{if } \theta_2 < 0 \\ [-4 - \lambda, -4 + \lambda] & \text{if } \theta_2 = 0 \\ 2\theta_2 - 4 + \lambda & \text{if } \theta_2 > 0 \end{cases}$$

$$\theta_2 = \begin{cases} 0 & \text{if } \lambda > 4 \\ \frac{4 - \lambda}{2} & \text{if } 0 < \lambda < 4 \end{cases}$$

We can now observe the following about θ:

- $\lambda > 4$; both θ_1 and θ_2 are 0.
- $1 < \lambda < 4$; then θ_2 is non-zero and θ_1 is 0.
- $0 < \lambda < 1$; both θ_1 and θ_2 are non-zero.

Question 3

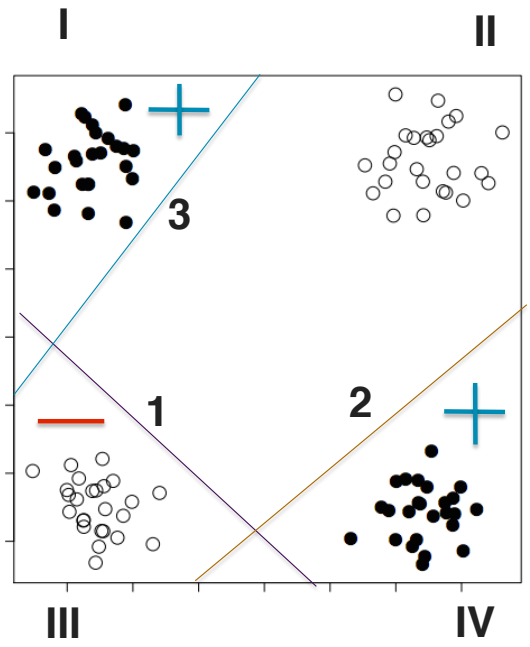

Figure 1: The figure illustrates how the weak classifiers are added to the data space and the decision boundary is built in every iteration.

The following are the steps involved in running the Adaboost algorithm on the dataset given in Figure 1. The steps described below follow Algorithm 16.2 from the course textbook, and note that we calculate $\alpha_m (= 2 \times \beta_m)$, at each step.
Initially, all the weights are $w_i = \frac{1}{100}$. In first step, the weak classifier ϕ_1 misclassifies

the points near region II; therefore, the error is

$$\begin{aligned} err_1 &= \frac{\sum w_i \mathbb{1}(y \neq \phi_1(x_i))}{\sum w_i} \\ &= 25 \times \frac{1}{100} = \frac{1}{4} \\ \alpha_1 &= \log \frac{1 - err_1}{err_1} \\ &= \log 3 \end{aligned}$$

The updated weights for these points are given by

$$\begin{aligned} w_i &= w_i \exp[\alpha_1 \mathbb{1}(y \neq \phi_1(x_i))] \\ w_i &= 3w_i \text{ for all } y \neq \phi_1(x_i) \\ w_i &= w_i \text{ for other points} \end{aligned}$$

Consequently, the renormalized weights are $w_i^{II} = \frac{3}{150}$ and other weights $w_i^{I,III,IV} = \frac{1}{150}$. Next, in step 2, the weak classifier ϕ_2 misclassifies the points in region I, now the error becomes

$$\begin{aligned} err_2 &= \frac{\sum w_i \mathbb{1}(y \neq \phi_2(x_i))}{\sum w_i} \\ &= 25 \times \frac{1}{150} = \frac{1}{6} \\ \alpha_2 &= \log \frac{1 - err_1}{err_1} \\ &= \log 5 \end{aligned}$$

The similar to the updates in the first step, the updated weights using the new α_2 are $w_i^I = \frac{5}{250}$, $w_i^{II} = \frac{3}{250}$, $w_i^{III} = \frac{1}{250}$ and $w_i^{IV} = \frac{1}{250}$
Finally, in step 3, the weak classifier ϕ_3 misclassifies region IV. Now, following the same steps as before we have $err_3 = \frac{1}{10}$; therefore, $\alpha_3 = \log 9$.
The final updated re-normalized weights are $w_i^I = \frac{5}{450}$, $w_i^{II} = \frac{3}{450}$, $w_i^{III} = \frac{1}{450}$ and $w_i^{IV} = \frac{9}{450}$. The final decision boundary is given by

$$f(x) = \text{sgn}\big(\log(3)\phi_1(x) + \log(5)\phi_2(x) + \log(9)\phi_3(x)\big).$$

Using more weak-classifiers, we can definitely build a more complex decision boundary, which improves on the accuracy and its slow to overfit.

For the points in region III, we have $f(x) = \text{sgn}(\log(3) - \log(5) - \log(9)) = \text{sgn}(-2.708) = -1$, which is true for the points in III. Similarly,

$$\text{for samples in region } IV, \quad f(x) = \text{sgn}(\log(3) + \log(5) - \log(9)) = \text{sgn}(0.510) = +1$$

$$\text{and samples in region } I, \quad f(x) = \text{sgn}(\log(3) - \log(5) + \log(9)) = \text{sgn}(1.686) = +1$$

Finally, for region II, we have $f(x) = \text{sgn}(-\log(3) - \log(5) - \log(9)) = \text{sgn}(-4.905) = -1$. As we can see the decision boundary learnt after 3 steps of Adaboost perfectly classifies the samples in Figure 1.