ETC3555

Practice exam 2018

Instructions

- In the **actual** exam, there are 7 questions worth a total of 100 marks. Each question will have between 4 and 6 subquestions. You should attempt them all.
- The exam is open book
- You can use the approved calculator

QUESTION 1

(a) True or false. The perceptron learning algorithm (PLA) will always decrease the number of misclassified examples after one update of the weights.

[2 marks]

False. An update of the PLA can increase the number of misclassified examples. But if the data is linearly separable, PLA is guaranteed to converge to a solution where all the examples are correctly classified.

(b) Briefly explain why the Hoeffding inequality does not apply to multiple bins.

[2 marks]

The hoeffding inequality bounds the probability that the absolute difference between the sample mean and the true mean is higher than a threshold. The sample mean is computed using one random sample of size N. With multiple bins, we have multiple samples. If we want to bound the probability that the absolute difference between the sample means and the true mean is higher than a threshold, we need to take into account the uncertainty associated to each random sample.

(c) In logistic regression, what is the hypothesis set if we consider inputs $\mathbf{x} \in \mathbb{R}^p$?

[2 marks]

In logistic regression, we consider the following hypotheses: $h(\mathbf{x}) = \theta(\mathbf{w}^T\mathbf{x})$ where $\mathbf{w} \in \mathbb{R}^p$ and $\theta(\cdot)$ is the logistic function.

(d) True or false. We can perform nonlinear regression using a regression learning algorithm that is linear in both the weights and the variables. Explain your answer.

[2 marks]

False. In that case, the learning algorithm reduces to linear regression. We can perform nonlinear regression with an algorithm that is linear in the weights but nonlinear in the variables.

(e) True or false. We can learn the XOR function with support vector machines.

[2 marks]

True. Any learning algorithm that can produce flexible (nonlinear) decision boundaries can learn the XOR function.

(f) True or false. A deep neural network with linear activation functions allows to model more complex target function. Explain your answer.

[2 marks]

False. With linear activation functions, the deep neural network can be written as a linear model (with more parameters).

[Total: 12 marks]

— END OF QUESTION 1 —

QUESTION 2

(a) We consider a hypothesis set with $M = 100$ different hypotheses. The learning algorithm picks the final hypothesis g using a dataset $\mathcal{D}$ with $N = 1000$ i.i.d. examples. Can we use the Hoeffding Inequality and state that

$$\mathbb{P}\left[|E_{\text{in}}(g) - E_{\text{out}}(g)| > \epsilon\right] \leq 2e^{-2000\epsilon^2}?$$

If the previous inequality is not valid, give the correct answer including explanations.

[4 marks]

No. This is exactly the "multiple bins/hypotheses" case. The Hoeffding Inequality is valid when the hypothesis is fixed. In this example, g is not fixed. It is selected among 100 hypotheses. The correct answer is given below. First, the event "$|E_{\text{in}}(g) - E_{\text{out}}(g)| > \epsilon$" implies $\cup_{m=1}^{100}$ "$|E_{\text{in}}(h_m) - E_{\text{out}}(h_m)| > \epsilon$" where $\cup$ is the union operator. Second, we can now use the fact that for any M events $\mathcal{B}_1, \mathcal{B}_2, \ldots, \mathcal{B}_M$, $\mathbb{P}(\cup_{m=1}^{M} \mathcal{B}_m) \leq \sum_{m=1}^{M} \mathbb{P}(\mathcal{B}_m)$. In other words, we have $\mathbb{P}\left[|E_{\text{in}}(g) - E_{\text{out}}(g)| > \epsilon\right] \leq \sum_{m=1}^{1} 00\mathbb{P}\left[|E_{\text{in}}(h_m) - E_{\text{out}}(h_m)| > \epsilon\right]$. We can now use the Hoeffding Inequality for each hypothesis h_m. This gives us $\mathbb{P}\left[|E_{\text{in}}(g) - E_{\text{out}}(g)| > \epsilon\right] \leq \sum_{m=1}^{100} \mathbb{P}\left[|E_{\text{in}}(h_m) - E_{\text{out}}(h_m)| > \epsilon\right] \leq \sum_{m=1}^{100} 2e^{-2000\epsilon^2} = 200e^{-2000\epsilon^2}$.

(b) Give the generalization bound associated to your answer in (a). More precisely, state that with probability at least 90%, "$E_{\text{out}}(g) \leq E_{\text{in}}(g) + \epsilon$". What is the value of ϵ?

[4 marks]

We have $\mathbb{P}\left[|E_{\text{in}}(g) - E_{\text{out}}(g)| > \epsilon\right] \leq 200\ e^{-2000\epsilon^2} \implies 1 - \mathbb{P}[|E_{\text{in}}(g) - E_{\text{out}}(g)| \leq \epsilon] \leq 200e^{-2\epsilon^2 1000} \implies \mathbb{P}[|E_{\text{in}}(g) - E_{\text{out}}(g)| \leq \epsilon] \geq 1 - 200e^{-2000\epsilon^2}$. In other words, with probability at least $1 - 200e^{-2000\epsilon^2}$, we have $|E_{\text{in}}(g) - E_{\text{out}}(g)| \leq \epsilon$, which implies $E_{\text{out}}(g) \leq E_{\text{in}}(g) + \epsilon$. We want $1 - 200e^{-2000\epsilon^2} = 0.90$, i.e. $\epsilon = \sqrt{\frac{1}{2000}\ ln\,(2000)} \approx 0.061$.

[Total: 8 marks]

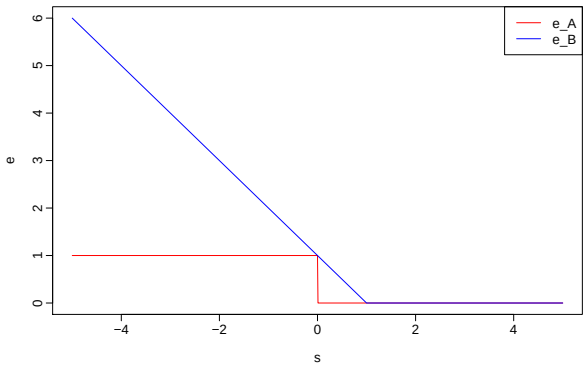
— END OF QUESTION 2 —

QUESTION 3

(a) Let $s = \mathbf{w}^T\boldsymbol{x}$, $\mathbf{w} \in \mathbb{R}^p$ and $y \in \{-1, 1\}$. Consider the following pointwise error measures: $e_A(s, y) = \mathbb{1}\{y \neq \text{sign}(s)\}$, and $e_B(s, y) = \max(0, 1 - ys)$.

(i) For $s \in [-5, 5]$ and $y = +1$, plot these two error measures on the same plot.

[2 marks]



(ii) Assume $y = +1$ and $\text{sign}(0) = -1$. Prove that $\forall s, e_A(s, y) \leq e_B(s, y)$.

[3 marks]

We want to show that $\mathbb{1}\{1 \neq \text{sign}(s)\} \leq \max(0, 1 - s)$. We will consider two cases: $s \leq 0$ and $s > 0$.

- If $s \leq 0$, we have $\max(0, 1 - s) = 1 - s \geq 1 = \mathbb{1}\{1 \neq -1\}$.
- If $s > 0$, we have $\max(0, 1 - s) = \begin{cases} 1 - s, & 0 < s \leq 1 \\ 0, & s > 1 \end{cases}$. If $0 < s \leq 1$, we have $1 - s \geq 0 = \mathbb{1}\{1 \neq +1\}$ and if $s > 1$, we have $0 \geq 0 = \mathbb{1}\{1 \neq +1\}$

(b) We consider the following classification learning algorithm. In each iteration t, pick a misclassified point $(\boldsymbol{x}_n, y_n)$, and compute $s(t) = \mathbf{w}^T(t)\ \boldsymbol{x}_n$. Update the weight $\mathbf{w}$ using $\mathbf{w}(t + 1) \leftarrow \mathbf{w}(t) + 0.1 \times (y_n - s(t))\boldsymbol{x}_n$. As far as classifying $\mathbf{x}_n$ is concerned, argue that the weight update moves the weights $\mathbf{w}(t)$ in the direction of classifying $\mathbf{x}_n$ correctly.

[4 marks]

If $\boldsymbol{x}_n$ is misclassified by $\mathbf{w}(t)$, then we have $y_n\mathbf{w}^T(t)\boldsymbol{x}_n < 0$. Furthermore, we have $y_n\mathbf{w}^T(t+1)\boldsymbol{x}_n = y_n(\mathbf{w}^T(t) + 0.1 \times (y_n - s(t))\boldsymbol{x}_n^T)\boldsymbol{x}_n = y_n\mathbf{w}^T(t)\boldsymbol{x}_n + \underbrace{y_n \times 0.1 \times (y_n - s(t))\boldsymbol{x}_n^T\boldsymbol{x}_n}_{\delta}$. We also have $\delta = 0.1 \times (y_n^2 - y_n s(t))\boldsymbol{x}_n^T\boldsymbol{x}_n = 0.1 \times (1 - y_n\mathbf{w}^T(t)\ \boldsymbol{x}_n)\boldsymbol{x}_n^T\boldsymbol{x}_n > 0$ since $y_n\mathbf{w}^T(t)\ \boldsymbol{x}_n < 0$ and $\boldsymbol{x}_n^T\boldsymbol{x}_n > 0$ (the first coordinate of $\boldsymbol{x}_n$ is 1). In other-words, a strictly positive quantity has been added to $y_n w^T(t)x_n$ (which is negative). As a result, $y_n w^T(t+1)x_n$ is now closer to being positive (which means a correct classification).

[Total: 9 marks]

— END OF QUESTION 3 —

QUESTION 4

(a) The error for a single data point $(\mathbf{x}_n, y_n)$ is given by $e_n(\mathbf{w}) = \max(0, 1 - y_n \mathbf{w}^T \mathbf{x}_n)^2$. Derive a learning algorithm to learn the weights $\mathbf{w}$ by applying gradient descent on $\mathrm{E_{in}}(\mathbf{w}) = \frac{1}{N} \sum_{n=1}^{N} e_n(\mathbf{w})$. You should give each step of your algorithm.

[4 marks]

The error function can be expressed as

$$e_n(\mathbf{w}) = \begin{cases} 0 & \text{if } y_n \mathbf{w}^T \mathbf{x}_n \geq 1, \\ (1 - y_n \mathbf{w}^T \mathbf{x}_n)^2 & \text{if } y_n \mathbf{w}^T \mathbf{x}_n < 1. \end{cases}$$

The derivative is given by

$$\nabla e_n(\mathbf{w}) = \begin{cases} 0 & \text{if } y_n \mathbf{w}^T \mathbf{x}_n \geq 1, \\ -2 y_n \mathbf{x}_n (1 - y_n \mathbf{w}^T \mathbf{x}_n) & \text{if } y_n \mathbf{w}^T \mathbf{x}_n < 1, \end{cases}$$

Data: $\{(\mathbf{x}_n, y_n)\}_{n=1}^{N}$
Result: final weights: $\mathbf{w}(t+1)$
initialise weights at $t = 0$ to $\mathbf{w}(0)$;
initialise the threshold ϵ ;
initialise the learning rate η ;
for $t = 0, 1, 2, \ldots$ **do**
  Compute the gradient

$$\nabla E_{\text{in}} = \frac{1}{N} \sum_{n=1}^{N} \nabla e_n(\mathbf{w}(t))$$

  update the weights: $\mathbf{w}(t+1) = \mathbf{w}(t) - \eta \, \nabla E_{\text{in}}$;
  iterate until $|\mathrm{E_{in}}(\mathbf{w}(t+1)) - \mathrm{E_{in}}(\mathbf{w}(t))| < \epsilon$;
end

(b) We want to apply mini-batch gradient descent using a dataset with N examples. To do so, we pick b examples (where $1 \leq b \leq N$), and apply gradient descent to these b examples. Explain how the value of b controls the bias and variance tradeoff in the gradient estimate, as well as the computational complexity of the procedure.

[4 marks]

The quantity we want to compute is the gradient on *all* examples. When $b = N$, this procedure reduces to gradient descent. With $b = 1$, it is equivalent to stochastic gradient descent. For any value of b, the estimate will be unbiased since we pick examples uniformly at random. The value of b will essentially control the variance. For small values of b, few examples are used and this can lead to high variance. However, a significant gain in computational complexity can be obtained especially when b is much smaller than N, and N is very large.

[Total: 8 marks]

— END OF QUESTION 4 —

QUESTION 5

(a) Let $\mathbf{x} = (1, x_1, x_2, x_3)^T$ where $x_j \in \{0, 1\}$ for $j = 1, 2, 3$. Consider a neural network without hidden layers and with an output layer with one unit where the activation function is the logistic function $\theta(s) = \frac{1}{1+\exp(-s)}$. Assign values to the weights of this network so that the output is greater than 0.5 if and only if $((x_1$ OR $x_2)$ AND $x_3)$ is true.

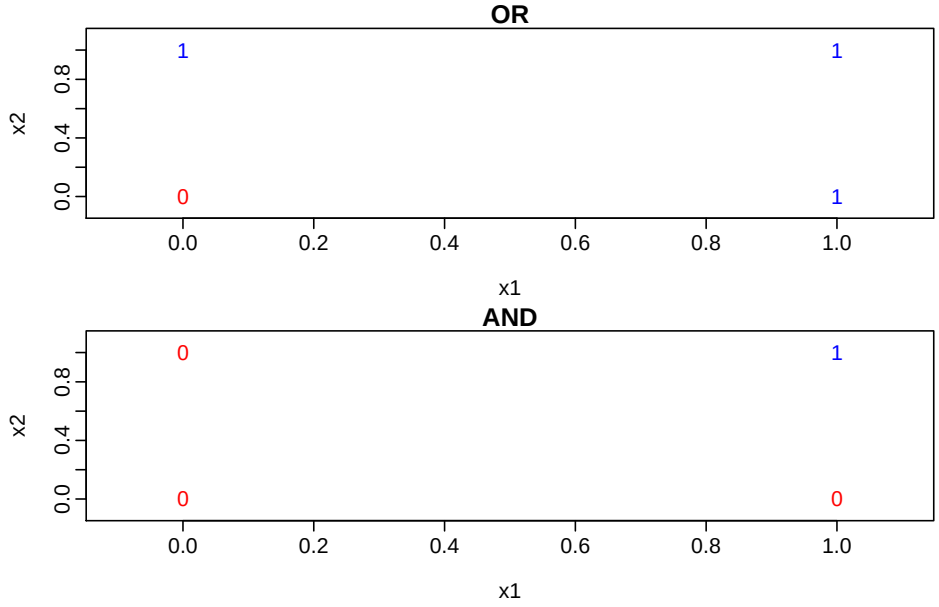
[4 marks]

We need to find w_0, w_1, w_2 and w_3 so that $\theta(w_0 + w_1x_1 + w_2x_2 + w_3x_3)$ returns a value greater than 0.5 if and only if $((x_1$ OR $x_2)$ AND $x_3)$ is true. Equivalently, we want $w_0 + w_1x_1 + w_2x_2 + w_3x_3$ to be larger than zero when $((x_1$ OR $x_2)$ AND $x_3)$ is true.

We can first compute the value of the function for all inputs.

x_1	x_2	x_3	y = $(x_1$ OR $x_2)$ AND x_3
0	0	1	0
0	1	1	1
1	0	1	1
1	1	1	1
0	0	0	0
0	1	0	0
1	0	0	0
1	1	0	0

Here we are dealing with binary variables (instead of positive and negative variables). We can draw the OR and AND functions to identify the hyperplane that separates the points, as can be seen in the following Figure.



For example, $\text{OR}(x_1, x_2) = \text{Threshold}(x_1 + x_2 - 0.6)$ and $\text{AND}(x_1, x_2) = \text{Threshold}(x_1 + x_2 - 1.5)$ where $\text{Threshold}(\cdot)$ is a function that returns 1 for a positive number and zero otherwise. By using these two observations, we can derive values of w_0, w_1, w_2 and w_3. One possible solution is $w_0 = -1$, $w_1 = w_2 = 0.5$, $w_3 = 1$. Note that there are multiple possibilities, and many ways to derive these weights. Any correct solution will satisfy $\theta(w_0 + w_1x_1 + w_2x_2 + w_3x_3) > 0.5$ if $y = 1$.

(b) The hyperbolic tangent is a soft threshold that can be used as activation function in neural networks. Show that the hard treshol dis an upper bound for the hyperbolic tangent.

[3 marks]

We have $\tanh(s) = \frac{e^s - e^{-s}}{e^s + e^{-s}}$. We want to show that $\tanh(s) < 1$ for $s > 0$, and $\tanh(s) > -1$ for $s < 0$. For $s < 0$, we have $\frac{e^s - e^{-s}}{e^s + e^{-s}} > -1 \implies e^s - e^{-s} > -e^s - e^{-s} \implies 2e^s > 0 \implies e^s > 0$, which is always true. For $s > 0$, we have $\frac{e^s - e^{-s}}{e^s + e^{-s}} < 1 \implies e^s - e^{-s} < e^s + e^{-s} \implies e^{-s} > 0$, which is always true.

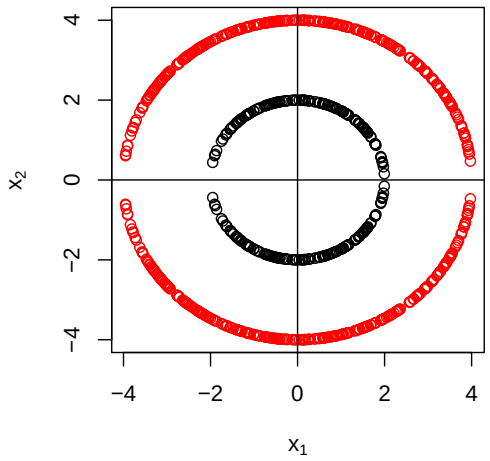
[Total: 7 marks]

— END OF QUESTION 5 —

QUESTION 6

(a) The following Figure shows a sample with 300 examples where the positive and negative examples are given in red and black, respectively. You compute your predictions using $\text{sign}(h(x_1, x_2))$ where $h(x_1, x_2) = w_0 + w_1 h_1(x_1) + w_2 h_2(x_2)$. In other words, the classifier is linear in the parameters. Give the values of w_0, w_1, w_2, as well as the functions h_1 and h_2 so that all the examples are classified correctly.

[3 marks]



The students should notice that this is a noiseless data that needs a nonlinear decision boundary, i.e. h_1 and h_2 should be nonlinear in the variables. One solution is $w_0 = -9$, $w_1 = 1$, $w_2 = 1$, $h_1(x_1) = x_1^2$, and $h_2(x_2) = x_2^2$.

(b) Let $x_i \in \mathbb{R}^p$ and $y_i \in \{-1, 1\}$ for $i = 1, \ldots, n$. We consider the following optimization problem:

$$\underset{\beta_0, \beta_1, \ldots, \beta_p, \epsilon_1, \ldots, \epsilon_n, M}{\text{maximize}} \; M$$

$$\text{subject to} \quad \sum_{j=1}^{p} \beta_j^2 = 1,$$

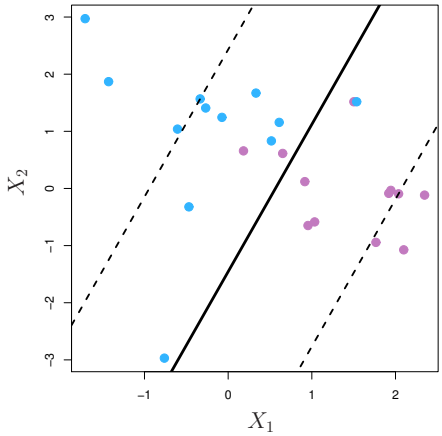
$$y_i(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \ldots + \beta_p x_{ip}) \geq M(1 - \epsilon_i),$$

$$\epsilon_i \geq 0, \quad \sum_{i=1}^{n} \epsilon_i \leq C,$$

where C is a nonnegative tuning parameter.

We solve the previous optimization problem for the dataset shown in the following Figure. Which value of the parameter C has been used: $C = 1$ or $C = 5$? Explain your answer.

[3 marks]



$C = 1$ cannot produce this separating hyperplane since we have 3 examples on the wrong side of the hyperplane. The value $C = 5$ has been used.

[Total: 6 marks]

— END OF QUESTION 6 —