Planning, Learning and Decision Making

MSc in Computer Science and Engineering

Second test – January 26, 2021

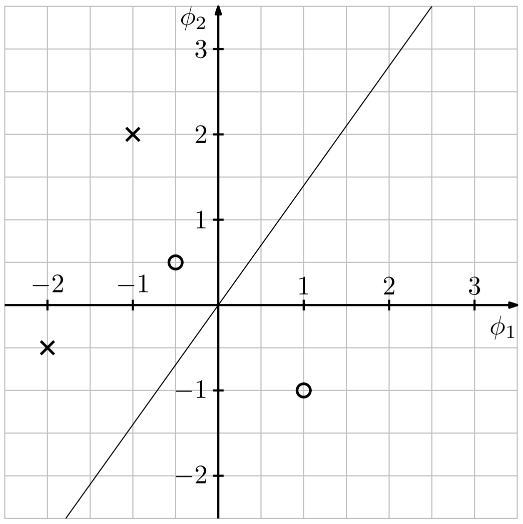
Version A

Instructions

- You have 90 minutes to complete the test.
- The test has a total of 5 questions, with a maximum score of 20 points. The questions have different levels of difficulty. The point value of each question is provided next to the question number.
- *If you get stuck in a question, move on.* You should start with the easier questions to secure those points, before moving on to the harder questions.
- *No interaction with the faculty is allowed during the exam.* If you are unclear about a question, clearly indicate it and answer to the best of your ability.
- Please provide your answers next to each question number, within the provided square brackets. For multiple-choice questions, provide only the letter corresponding to your option.
- For open answer questions, provide your answer succinctly the square brackets. If you feel the need to use math, avoid complicated notation, and write down the names of complicated symbols. You can also use programming-like notation to indicate computations.
- The exam is open book and open notes, and you may consult any written or printed material. You can also use a calculator. The consultation or use of any other type of electronic or communication equipment during the test is not allowed.
- Good luck.

Question 1. (6 pts.)

Consider the dataset depicted in the grid below, where each data-point is described by two features, ϕ_1 and ϕ_2. The points marked with "$\times$" correspond to class 1, and the points marked with "$\circ$" correspond to class 0.



Consider a logistic regression classifier, where

$$y = \hat{\pi}(1 \mid x) \triangleq \mathbb{P}\left[\mathrm{a} = 1 \mid \mathrm{x} = x\right] = \frac{1}{1 + \exp(-w_0 - w_1\phi_1(x) - w_2\phi_2(x))},$$

and suppose that

$$w_0 = 0.0; \qquad w_1 = -1.4; \qquad w_2 = 1.0.$$

(a) **(1 pt.)** Is the dataset linearly separable? Justify your answer.

(b) **(1 pt.)** The decision boundary for the classifier is described by the expression (select the *only* correct answer):

A. $\phi_2(x) = 1.4\phi_1(x)$.

B. $\phi_2(x) = -1.4\phi_1(x)$.

C. $\phi_2(x) + \phi_1(x) = 0$.

D. None of the above.

(c) **(2.5 pts.)** For each of the points in the above plot, compute the corresponding class, and indicate the accuracy of the classifier in that dataset.

(d) **(2.5 pts.)** Logistic regression classifiers can be trained using stochastic gradient descent to minimize the empirical risk

$$\hat{L}_N(\hat{\pi}) = \sum_{n=1}^{N} a_n \log \hat{\pi}(1 \mid x_n) + (1 - a_n) \log(1 - \hat{\pi}(1 \mid x_n)).$$

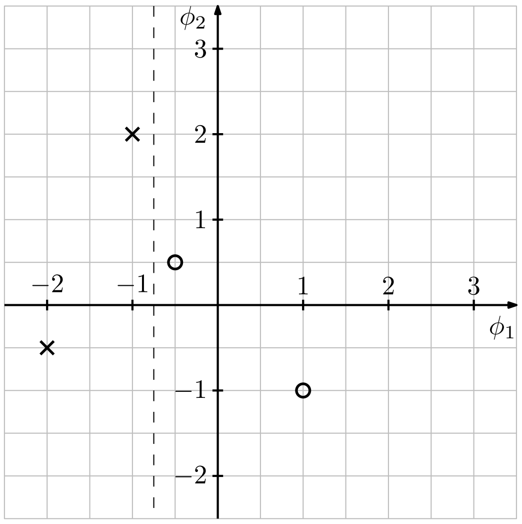
For a single training point (x, a), the gradient is given by

$$\mathbf{g} = \phi(x)(\hat{\pi}(1 \mid x) - a).$$

Suppose then that you are given a training point (x, a) such that $\phi_1(x) = 2.0$, $\phi_2 = 2.8$, and $a = 1$. Perform a stochastic gradient descent update to the weights of the classifier, using a step-size $\alpha = 1$.

Solution 1.

(a) The dataset is linearly separable. For example, a vertical line between the two datapoints in the same quadrant linearly separates the points belonging to the two classes (see diagram below).



(b) A ($\phi_2(x) = 1.4\phi_1(x)$).

(c) The decision boundary is depicted in the original grid. We can easily check that

$x = (1.0, -1.0)$	$w_1\phi_1(x) + w_2\phi_2(x) = -1.4 - 1 = -2.4 < 0$	(class 0)
$x = (-0.5, 0.5)$	$w_1\phi_1(x) + w_2\phi_2(x) = 0.7 + 0.5 = 1.2 > 0$	(class 1)
$x = (-2.0, -0.5)$	$w_1\phi_1(x) + w_2\phi_2(x) = 2.8 - 0.5 = 2.3 > 0$	(class 1)
$x = (-1.0, 2.0)$	$w_1\phi_1(x) + w_2\phi_2(x) = 1.4 + 2 = 3.4 > 0$	(class 1)

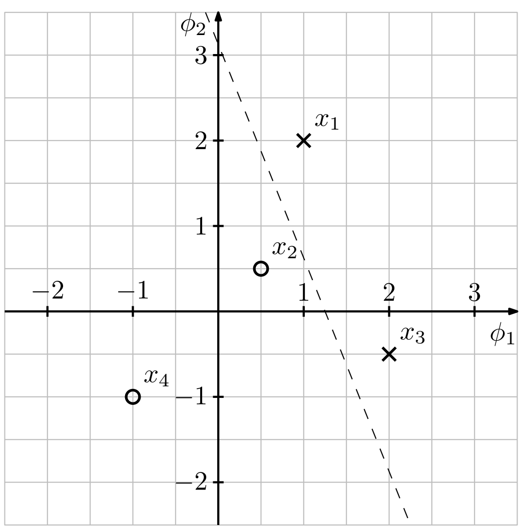
The accuracy is 75%.

(d) We have, for the provided training point, $\pi(1 \mid x) = 0.5$. Then,

$$w_0 \leftarrow w_0 - \alpha(\hat{\pi}(1 \mid x) - a) = 0 - 1 \times (0.5 - 1) = 0.5.$$
$$w_1 \leftarrow w_1 - \alpha\phi_1(x)(\hat{\pi}(1 \mid x) - a) = -1.4 - (2.0) \times (0.5 - 1) = -0.4.$$
$$w_2 \leftarrow w_2 - \alpha\phi_2(x)(\hat{\pi}(1 \mid x) - a) = 1.0 - (2.8) \times (0.5 - 1) = 2.4.$$

Question 2. (**2 pts.**)

Consider a binary classification problem, where the data is once again described by two features, ϕ_1 and ϕ_2. Consider the dataset represented in the following grid, where the points marked with "$\times$" correspond to class 1, and the points marked with "$\circ$" correspond to class -1.



The black dashed line corresponds to the max-margin classifier, computed by solving the constrained optimization problem

$$\begin{aligned} &\min && \|w\|^2 \\ &\text{s.t.} && a_n(w_0 + w_1\phi_1(x_n) + w_2\phi_2(x_n)) \geq 1, \text{for all training points } (x_n, a_n). \end{aligned}$$

(a) **(1 pt.)** The set of support vectors for the classifier is (select the *only* correct answer):

A. $\{x_1, x_3\}$.

B. $\{x_1, x_3, x_4\}$.

C. $\{x_2, x_3\}$.

D. None of the above.

(b) **(1 pt.)** Consider, in particular, the point x_1. It holds that (select the *only* correct answer):

A. $w_0 + w_1\phi_1(x_1) + w_2\phi_2(x_1) = 0$.

B. $w_0 + w_1\phi_1(x_1) + w_2\phi_2(x_1) = 1$.

C. $w_0 + w_1\phi_1(x_1) + w_2\phi_2(x_1) = -1$.

D. None of the above.

Solution 2.

(a) D (None of the above. The support vectors are $\{x_1, x_2, x_3\}$).

(b) B ($w_0 + w_1\phi_1(x_1) + w_2\phi_2(x_1) = 1$).

Question 3. (2 pts.)

In *batch* reinforcement learning algorithms like fitted Q-iteration, the agent builds a "dataset" of transitions, $\mathcal{D} = \{(x_n, a_n, c_n, x'_n), n = 1, \ldots, N\}$ and computes, at each iteration t, a new approximation $\hat{Q}_{w_t}$, where

$$w_t = \underset{w}{\operatorname{argmin}} \sum_{n=1}^{N} (c_n + \gamma \min_{a \in \mathcal{A}} \hat{Q}_{w_{t-1}}(x'_n, a) - \hat{Q}_{w_t}(x_n, a_n))^2. \tag{1}$$

Consider the particular case where fitted Q-iteration is run with a linear approximation, i.e.,

$$\hat{Q}_w(x, a) = \phi^\top(x, a)w,$$

for some features ϕ. Further suppose that fitted Q-iteration performs the minimization in (1) by following stochastic gradient descent, performing a gradient descent update for every point in $\mathcal{D}$. In other words, each iteration of fitted Q-iteration would perform N updates to w_t using the update rule

$$w_t \leftarrow w_t + \alpha\phi(x_n, a_n)(c_n + \gamma \min_{a \in \mathcal{A}} \hat{Q}_{w_{t-1}}(x'_n, a) - \hat{Q}_{w_t}(x_n, a_n)). \tag{2}$$

Can you identify any difference between the update (2) and the standard update for Q-learning with linear function approximation? Can such differences explain why fitted Q-iteration is more stable than Q-learning with linear function approximation?

Solution 3.

The key difference is in the target for the update. While Q-learning would use the current estimate, $\hat{Q}_{w_t}$, to compute the target $c_n + \gamma \min_{a \in \mathcal{A}} \hat{Q}_{w_t}(x'_n, a)$, fitted Q-iteration instead uses the *previous* estimate, $\hat{Q}_{w_{t-1}}$. This means that, as the weights w_t are optimize in a step of FQI, the target remains constant (i.e., these updates do not bootstrap), which improves stability of the updates, since bootstrapping is one of the elements that causes divergence of Q-learning with function approximation.

Question 4. (6 pts.)

Consider an MDP $(\mathcal{X}, \mathcal{A}, \{\mathbf{P}_a\}, c, \gamma)$, where $\mathcal{X} = \{1, 2, 3\}$, $\mathcal{A} = \{A, B\}$, and $\gamma = 0.9$. Suppose that the agent is following an ε-greedy policy that selects an exploratory action with probability ε and a greedy action with probability $1 - \varepsilon$.

Suppose that, at time step t, the agent's current estimate for the Q-function is

$$\hat{\mathbf{Q}} = \begin{bmatrix} 0.30 & 0.20 \\ 0.27 & 0.21 \\ 0.0 & 0.01 \end{bmatrix}.$$

(a) **(1.5 pts.)** Compute the greedy policy with respect to $\hat{Q}$.

(b) **(1.5 pts.)** Suppose that, at time step t, $x_t = 1$ and the agent selects a greedy action. It observes a cost $c_t = 1.0$ and transitions to state $x_{t+1} = 1$. Using a step-size $\alpha = 0.1$, compute $\hat{Q}$ after one update of Q-learning.

(c) **(1.5 pts.)** Suppose once again that, at time step t, $x_t = 1$ and the agent selects a greedy action. It observes a cost $c_t = 1.0$ and transitions to state $x_{t+1} = 1$, after which it performs an exploratory (non-greedy) action. Using a step-size $\alpha = 0.1$, compute $\hat{Q}$ after one update of SARSA. **Note:** You should use the original estimate $\hat{Q}$, not the one you computed in (b).

(d) **(1.5 pts.)** Comment on the differences between the updates in (b) and (c).

Solution 4.

(a) The greedy policy is given by

$$\pi_g = \begin{bmatrix} 0.0 & 1.0 \\ 0.0 & 1.0 \\ 1.0 & 0.0 \end{bmatrix}.$$

(b) If the agent selects a greedy action at time step t, the observed transition is $(1, B, 1.0, 1)$. The corresponding Q-learning update thus comes

$$\begin{aligned} Q(1, B) &= Q(1, B) + \alpha(c_t + \gamma \min_{a \in \mathcal{A}} Q(1, a) - Q(1, B)) \\ &= 0.2 + 0.1(1 + 0.9 \times 0.2 - 0.2) = 0.298. \end{aligned}$$

(c) If the agent selects a greedy action at time step t and an exploratory, non-greedy action at time step $t + 1$, the observed transition is $(1, B, 1.0, 1, A)$. The corresponding SARSA update thus comes

$$\begin{aligned} Q(1, B) &= Q(1, B) + \alpha(c_t + \gamma Q(1, A) - Q(1, B)) \\ &= 0.2 + 0.1(1 + 0.9 \times 0.3 - 0.2) = 0.307. \end{aligned}$$

(d) Q-learning is an *off-policy* algorithm. In computing the target, $c_t + \gamma \min_{a \in \mathcal{A}} Q(x_{t+1}, a)$, it uses as the next action that of a policy different from the one that the agent is actually using (the greedy action). In our case, it uses the action B (the greedy action). SARSA, on the other hand, is an *on-policy* algorithm. In computing the target, $c_t + \gamma Q(x_{t+1}, a_{t+1})$, it uses as the next action that of the agent's current policy. In our case, it uses the action A, which was selected by the ε-greedy policy.

Question 5. (4 pts.)

For each of the following questions, select the *single* most correct answer.

(a) **(1 pt.)** Let R_T denote the regret of a sequential prediction algorithm $\mathfrak{A}$ at time step T. The algorithm $\mathfrak{A}$ is *no regret* if and only if:

A. $R_T \to 0$.

B. $R_T / T \to 0$.

C. $R_T = 0$.

D. None of the above.

(b) **(1 pt.)** Out of all the multi-armed bandit algorithms discussed in the course, the one most suited to address *stochastic* multi-armed bandit problems is

A. Weighted majority.

B. EWA.

C. EXP3.

D. None of the above.

(c) **(1 pt.)** The weighted majority algorithm is most suited to address sequential prediction problems in which...

A. ... the predictions are binary.

B. ... the agent can only observe the cost for the action it selects.

C. ... the agent wants to minimize the total sum of discounted costs.

D. None of the above.

(d) **(1 pt.)** The EXP3 update is similar to that of EWA, the main difference being

A. ... the latter is designed for multi-armed bandit problems.

B. ... the latter must compensate for the fact that it can only update a single weight.

C. ... the update is not multiplicative.

D. None of the above.

Solution 5.

(a) B ($R_T/T \to 0$).

(b) D (None of the above. It's actually the UCB algorithm).

(c) A (... the predictions are binary).

(d) D (None of the above. Actually, the former must compensate for the fact that it can only update a single weight, not the other way around).