CS181 Practice Questions: Support Vector Machines and MDPs

1. **Weights of Hard Margin SVM (MIT 6.867, Fall '12)**

   Consider a hard-margin SVM classification scenario where we have linearly separable data $\{\boldsymbol{\phi}_n, t_n\}_{n=1}^N$, where $t_n \in \{-1, -1\}$. As usual, for a new data-point $\boldsymbol{\phi}_{new}$, we predict 1 if $\boldsymbol{w}^\top \boldsymbol{\phi}_{new} + b \geq 0$, and -1 otherwise. Now, assume $N = 4$, and that our data is $\boldsymbol{\phi}_1 = [1, 1]^\top$, $\boldsymbol{\phi}_2 = [2, 2]^\top$, $\boldsymbol{\phi}_3 = [-1.5, -1.5]^\top$, $\boldsymbol{\phi}_4 = [4, 4]^\top$. Show that for any labeling $t_1, \dots, t_4$ of our 4 data-points, the $\boldsymbol{w}^\star$ we learn by optimizing the hard-margin SVM criterion will have $w_1^\star = w_2^\star$.

   As in Bishop (7.8), by taking the derivative of the primal hard-margin SVM objective wrt $\boldsymbol{w}$, setting to 0, and solving for $\boldsymbol{w}$, we have that $\boldsymbol{w}^\star = \sum_{n=1}^N a_n t_n \boldsymbol{\phi}_n$. Since all of our $\boldsymbol{\phi}_n$ have the same first and second coordinate, $\boldsymbol{w}^\star$, which is a linear combination of the $\boldsymbol{\phi}_n$, will too.

2. **SVM with Only Positive Training Examples (MIT 6.867, Fall '12)**

Suppose we have data $D = \{\boldsymbol{\phi}_n, t_n\}_{n=1}^N$, where $t_n \in \{-1, -1\}$, and that we would like to learn a linear classification boundary by optimizing an SVM-like criterion that completely ignores the negative training examples. In particular, letting $D^+ = \{\boldsymbol{\phi}_n \in D \mid t_n = 1\}$ (i.e., the set of positive training examples), we will look for

$$\arg\min_{\boldsymbol{w}} \frac{1}{2}||\boldsymbol{w}||^2 \quad \text{s.t.} \quad \boldsymbol{w}^\top \boldsymbol{\phi}_i \geq 1, \forall\, \boldsymbol{\phi}_i \in D^+. \tag{1}$$

Note that the above optimization problem does not include an offset parameter b!

(a) If instead our optimization problem above *did* include an offset parameter b (i.e., we minimized $||\boldsymbol{w}||^2$ subject to $\boldsymbol{w}^\top \boldsymbol{\phi}_i + b \geq 1$, $\forall\, \boldsymbol{\phi}_i \in D^+$), what would the minimizing weight-vector $\boldsymbol{w}^\star$ be?

(b) If we've found a $\boldsymbol{w}^\star$ that minimizes Equation (1), what is the value of $\min_{\boldsymbol{\phi}_i \in D+} \boldsymbol{w}^{\star\top} \boldsymbol{\phi}_i$?

(c) Again, assuming we've found a $\boldsymbol{w}^\star$ that minimizes Equation (1), suppose that for a new data-point $\boldsymbol{\phi}_{new}$ we predict 1 if $\boldsymbol{w}^{\star\top} \boldsymbol{\phi}_{new} \geq (\min_{\boldsymbol{\phi}_i \in D+} \boldsymbol{w}^{\star\top} \boldsymbol{\phi}_i) - \epsilon$ for some small $\epsilon > 0$, and -1 otherwise. Will this decision rule guarantee that all the training examples in D (both positive and negative) are classified correctly?

(a) We would have $\boldsymbol{w}^\star = \mathbf{0}$ (i.e., the zero-vector), since we can satisfy the constraints by just choosing b to be ≥ 1.

(b) 1. (Otherwise, we can make $\boldsymbol{w}^\star$ smaller).

(c) No, it won't. In particular, our optimization pays no attention to where the negative examples lie, and they could be on either side of the learned hyperplane.

3. **SVM with Three Points (MIT 6.867, Fall '12)**

Consider the following dataset consisting of three points on the real line

$$(x_1 = -1, t_1 = 1), (x_2 = 0, t_2 = -1), (x_3 = 1, t_3 = 1),$$

which we will attempt to separate with a linear hyperplane through feature-space *that goes through the origin*. That is, our discriminant function will be $\boldsymbol{w}^\top \phi(x) \geq 0$, with no offset term b. The primal form of this problem is:

$$\begin{aligned}
\text{Minimize} \quad & \frac{1}{2}||\boldsymbol{w}||^2 + C\sum_{n=1}^{3} \xi_n \\
\text{s.t.} \quad & t_n(\boldsymbol{w}^\top \phi(x_n)) \geq 1 - \xi_n, \quad \xi_n \geq 0, \quad i = 1, 2, 3
\end{aligned}$$

The dual form is:

$$\begin{aligned}
\text{Maximize} \quad & \sum_{n=1}^{3} a_n - \frac{1}{2}\sum_{n=1}^{3}\sum_{m=1}^{3} a_n a_m t_n t_m k(x_n, x_m) \\
\text{s.t.} \quad & 0 \leq a_n \leq C, \quad i = 1, 2, 3
\end{aligned}$$

(a) Suppose our kernel function is $k(x, x') = 1 + |xx'|$, where $|\cdot|$ is the absolute value. What feature mapping ϕ results in this kernel function?

(b) Using the feature mapping ϕ from your answer to the previous question, are the three training examples linearly separable in feature space?

(c) Consider any pair of kernels $k_1(x, x')$ and $k_2(x, x')$ such that the training points are linearly separable with k_1 but not with k_2. Are the data linearly separable if we use the kernel $k(x, x') = k_1(x, x') + k_2(x, x')$? Justify your answer.

(d) Using the kernel $k(x, x') = 1 + |xx'|$ that we had in part (a), express the value of $\boldsymbol{w}^\top \phi(x_2)$ in terms of the a_n (i.e., in dual form).

(e) If in our optimization we set $C < 1$, will we necessarily have $\xi_2 > 0$ (i.e., will the slack variable for the second training example exceed 0)? (Hint: you will probably want to use the expression you derived in the previous question for $\boldsymbol{w}^\top \phi(x_2)$ in answering this).

(a) $\phi(x) = \langle 1, |x| \rangle$.

(b) Yes.

(c) Yes, they are. In particular, adding kernels corresponds to concatenating the feature vectors induced by their corresponding ϕ functions. If the data are linearly separable in the ϕ_1 representation, we can at worst give 0 weight to the features from the ϕ_2 representation and still separate the data.

(d) By Bishop (7.29), we have

$$\boldsymbol{w}^{\mathsf{T}}\phi(x) = \sum_{n=1}^{3} a_n t_n k(x, x_n) = a_1 t_1 k(0,-1) + a_2 t_2 k(0,0) + a_3 t_3 k(0,1) = a_1 - a_2 + a_3$$

(e) Our optimization problem requires that $t_2(\boldsymbol{w}^{\mathsf{T}}\phi(x_2)) \geq 1 - \xi_2$, which by part (d) implies that we require $-1(a_1 - a_2 + a_3) = a_2 - a_1 - a_3 \geq 1 - \xi_2$. Since we set $C < 1$, and we know that all a_n must satisfy $0 \leq a_n \leq C < 1$, we will only be able to have $a_2 - a_1 - a_3 \geq 1 - \xi_2$ if $\xi_2 > 0$.

4. **(Berkley, Fall '11)**

> Recall that the equation of an ellipse in the 2-dimensional plane is $c(x_1 - a)^2 + d(x_2 - b)^2 - 1 = 0$. Show that an SVM using the polynomial kernel of degree 2, $K(\boldsymbol{u}, \boldsymbol{v}) = (1 + \boldsymbol{u} \cdot \boldsymbol{v})^2$ is equivalent to a linear SVM in the feature space $(1, x_1, x_2, x_1^2, x_2^2, x_1 x_2)$. This shows that SVM's with this kernel can separate any elliptic region from the rest of the plane.

The ellipse equation expands into the terms

$$cx_1^2 + dx_2^2 - 2acx_1 - 2bdx_2 + (a^2c + b^2d - 1) = 0$$

so we can view it as $\boldsymbol{w} = (x, d, -2ac, -2bd)$ with intercept $a^2c + b^2d - 1$ using the feature space $(1, x_1, x_2, x_1^2, x_2^2, x_1 x_2)$. Thus, in this feature space the elliptical boundary is linear, so we have linear separability.

5. **Value Iteration**

Say an MDP has state space S and reward R where all rewards are positive. If you run value iteration, what is the largest k for which all $V_k(s)$ are zero?

$k = 0$. $V_0(s) = 0$ for all states s. Since $V_1(s) = max\{r_1, r_2...\}$ and all rewards are positive, all $V_k(s) > 0$ when $k > 0$.

6. **Infinite Horizon**

> You are on a linear space and can move only right, left, or stay in the same place. Each position has reward r_i and $\gamma \approx 1$. Describe your optimal policy given any state i (don't forget about ties).

You always move towards state i that has $r_i(s) = max\{R\}$ and stay there forever. When there are ties, you move to the closest state i and then stay there. "Closest" is calculated as the state k with max $\sum r_j$ where $j \in [i, i+1, ..., k]$.

7. **Value Iteration vs Expectimax Search**

> What is the running time of Expectimax Search and Value Iteration as a function of the horizon, T, the number of actions, M, the the number of transitions for any state and action, L, and the number of steps, N? Why don't we always choose the algorithm with better asymptotic running time?

Expectimax Search takes time proportional to the height of the game tree (see Figure 3 in the lecture notes on MDP's), so we see that the running time is $O((ML)^T)$. Value Iteration uses dynamic programming to ensure that it doesn't revisit the same state multiple times, and we get running time linear in the inputs $O(NMLT)$. Although value iteration is a linear-time algorithm, it visits states that might not actually be reachable, while Expectimax only runs computations on states that can be reached. For some problem domains, this results in higher efficiency.

8. **Expected Utility and Optimal Actions**

Write the equation to compute the expected utility of taking an action a when in state s and the condition to determine the optimal action from the expected utility.

The expected utility of an action can be written as

$$\sum_{s' \in S} P(s'|s.a)U(s')$$

and we can compute the optimal action by solving the following

$$\arg\min_{a' \in A} \sum_{s' \in S} P(s'|s,a)U(s')$$

9. **Setting Rewards to Compute Shortest Path**

Suppose we have a grid-world where each state is represented as a point in $\{(x, y) | 0 \le x \le 4, 0 \le y \le 4\}$. Suppose you have a robot that starts in the lower left corner and is given a goal point that is needs to reach. At each state, the robot can move one point to the left, right, up, or down, and each action has a 90% chance of success (otherwise you stay at the current point).

- What is the size of the state space in the resulting MDP?
- Suppose we want to minimize the length of path to the goal, but we have no preference for which path the robot should take (and all points are reachable). What rewards could we assign to each state so that we recover a shortest path to the goal as our optimal policy?

- Our grid is composed of each point in a 5×5 grid, so we have 25 possible states in the state-space.
- We can set the reward of each state, new state pair (that does not end with the goal node) to be the same negative value. Then, when we try to maximize our reward we will choose the shortest path, and since each transition has the same reward, we do not preferentially choose any paths with the same length.