10-701 Machine Learning, Spring 2011: Homework 6

Instructions This homework is completely optional, and will NOT be collected or graded. We provide it to help you review the final exam. Feel free to work on this together with other students as you study for the exam.

1 Kernel and SVM

1. **Kernel**: Kernel functions implicitly define some mapping function $\phi(\cdot)$ that transforms an input instance $\mathbf{x} \in \mathbb{R}^d$ to high dimensional space Q by giving the form of dot product in Q: $K(\mathbf{x}_i, \mathbf{x}_j) \equiv \langle \phi(\mathbf{x}_i) \ , \ \phi(\mathbf{x}_j) \rangle$.

   (a) Prove that the kernel is symmetric, i.e. $K(\mathbf{x}_i, \mathbf{x}_j) = K(\mathbf{x}_j, \mathbf{x}_i)$.

   (b) Assume we use radial basis kernel function $K(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\frac{1}{2}\|\mathbf{x}_i - \mathbf{x}_j\|^2\right)$. Thus there is some implicit unknown mapping function $\phi(\mathbf{x})$. Prove that for any two input instances $\mathbf{x}_i$ and $\mathbf{x}_j$, the squared Euclidean distance of their corresponding points in the feature space Q is less than 2, i.e. prove that $\|\phi(\mathbf{x}_i) - \phi(\mathbf{x}_j)\|^2 \leq 2$.

2. **SVM**: With the help of a kernel function, SVM attempts to construct a hyper-plane in the feature space Q that maximizes the margin between two classes. The classification decision of any $\mathbf{x}$ is made on the basis of the sign of

$$\langle \hat{\mathbf{w}} \ , \ \phi(\mathbf{x}) \rangle + \hat{w}_0 = \sum_{i \in SV} y_i \alpha_i K(\mathbf{x}_i, \mathbf{x}) + \hat{w}_0 = f(\mathbf{x}; \alpha, \hat{w}_0),$$

   where $\hat{\mathbf{w}}$ and $\hat{w}_0$ are parameters for the classification hyper-plane in the feature space Q, SV is the set of support vectors, and α_i is the coefficient for the i-th support vector. Again we use the radial basis kernel function. Assume that the training instances are linearly separable in the feature space Q, and assume that the SVM finds a margin that perfectly separates the points.

   If we choose a test point $\mathbf{x}_{far}$ which is far away from any training instance $\mathbf{x}_i$ (distance here is measured in the original space $\mathbb{R}^d$), prove that $f(\mathbf{x}_{far}; \alpha, \hat{w}_0) \approx \hat{w}_0$.

2 Active Learning

This simple question tests our understanding on uncertainty sampling and version space reduction as two popular active learning strategies. Recall in the class that Burr used the example of learning the threshold to classify between poisonous and safe fruits, as shown in the following figure.

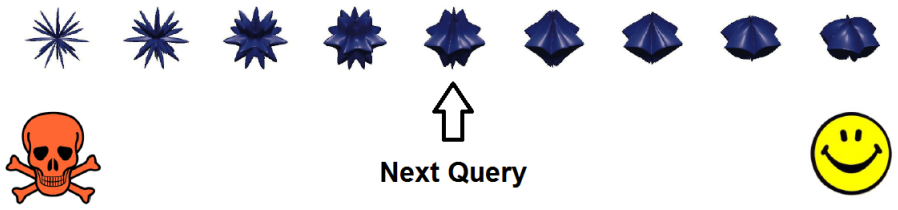


Figure 1: Learning the threshold between poisonous and safe fruits, with two labeled examples.

In the figure we already have two labeled examples: the leftmost one is poisonous ($y = +$) and the rightmost one is safe ($y = -$). Now we want to decide which fruit (among the pool of seven unlabeled fruits in the figure) to query next. According to binary search, we should query the label of the fruit in the middle.

2.1 Uncertainty sampling

Justify this choice from the perspective of uncertainty sampling. To do this, we need: 1) choose a class of hypothesis (i.e., a type of classifiers) we want to learn; 2) point out what is the classifier we will learn given the two labeled examples; 3) select a proper measure of uncertainty (three taught in the class); 4) given the learned classifier and the uncertainty measure, argue why we should choose the fruit in the middle.

2.2 Version space reduction

Justify the choice of querying the fruit in the middle from the perspective of version space reduction.

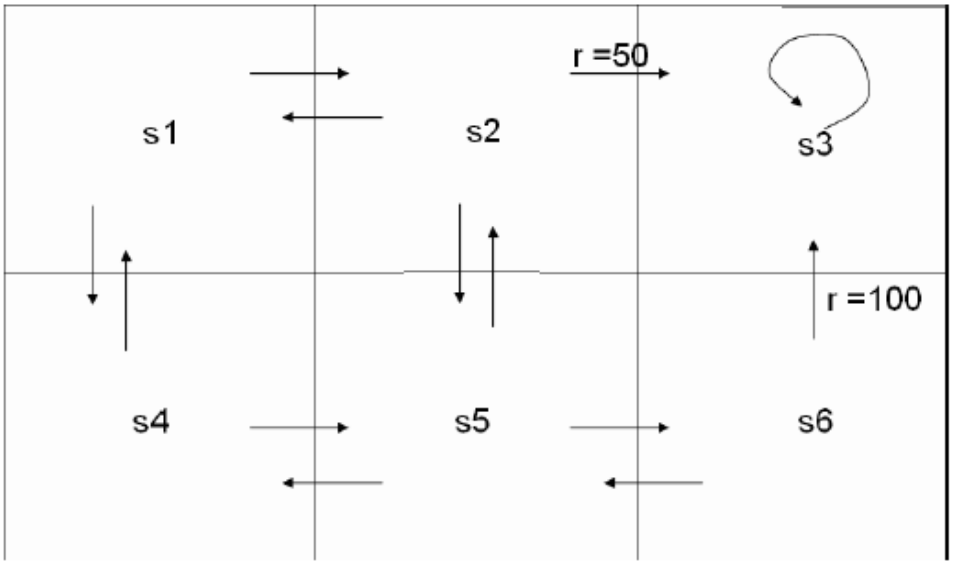
Hint 1. We can assume the entire hypothesis space contains only a finite number of classifiers: since there are 9 fruits in total, there are 10 possible classifiers (each classifier is a threshold such that all fruits on the left side are poisonous and all fruits on the right side are safe).

Hint 2. When we choose a fruit to query, its label is unknown. However, the reduction of the version space depends on the actual label of the queried fruit. As a result, we can choose an unlabeled fruit to maximize the "expected" reduction of the version space (expectation taken over the distribution of this fruit's label), to maximize the "best-case" reduction of the version space (i.e., when the fruit's label is the one that maximizes the version space reduction), or to maximize the "worst-case" reduction of the version space (i.e., when the fruit's label is the one that minimizes the version space reduction). We need to choose this notion properly to justify the choice of querying the fruit in the middle.

3 MDPs and Reinforcement Learning

Part A.:

Consider the following deterministic Markov Decision Process (MDP), describing a simple robot grid world. Notice the values of the immediate rewards are written next to transitions. Transitions with no value have an immediate reward of 0. **Assume the discount factor** $\gamma = 0.8$.



1. For each state s, write the value for $V^*(s)$ inside the corresponding square in the diagram.
2. Mark the state-action transition arrows that correspond to one optimal policy. If there is a tie, always choose the state with the smallest index.
3. Give a different value forγ which results in a different optimal policy and the number of changed policy actions should be minimal. Give your new value for γ and describe the resulting policy by indicating which $\pi(s)$ values (i.e., which policy actions) change.

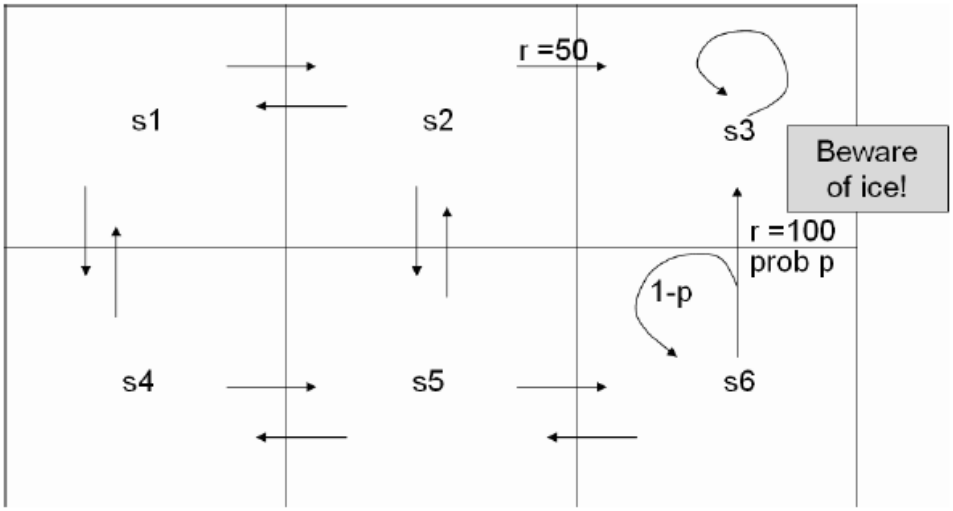
   New value for γ:

   Changed policy actions:

   For the remainder of this question, assume again that $\gamma = 0.8$.
4. How many complete loops (iterations) of value iteration are sucient to guarantee finding the optimal policy for this MDP? Assume that values are initialized to zero, and that states are considered in an arbitrary order on each iteration.
5. Is it possible to change the immediate reward function so that V^* changes but the optimal policy π^* remains unchanged? If yes, give such a change, and describe the resulting change to V^*. Otherwise, explain in at most 2 sentences why this is impossible.

Part B.:

It is December. Unfortunately for our robot, a patch of ice has appeared in its world, making one of its actions non-deterministic. The resulting MDP is shown below. Note that now the result of the action "go north" from state $s6$ results in one of two outcomes. With probability p the robot succeeds in transitioning to state $s3$ and receives immediate reward 100. However, with probability $(1-p)$, it slips on the ice, and remains in state $s6$ with zero immediate reward. **Assume the discount factor** $\gamma = 0.8$.



1. Assume $p = 0.7$. Write in the values of V^* for each state, and circle the actions in the optimal policy.
2. How bad does the ice have to get before the robot will prefer to completely avoid it? Answer this question by giving a value for p below which the optimal policy chooses actions that completely avoid the ice, even choosing the action "go west" over "go north" when the robot is in state $s6$.