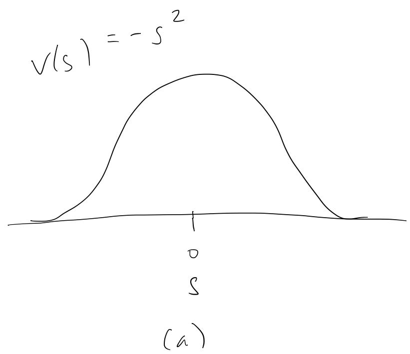
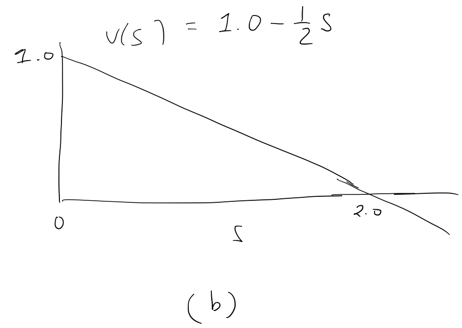
Worksheet 11

1. Consider the following two functions.





(a) Design features for each function, to approximate them as a linear function of these features. Can you design features to make the approximation exact?

(b) Can you design one set of features, that allows you to represent both functions?

Answer:

(a) For function a, we use one feature: s^2. Using this feature we can make an exact approximation with a weight of -1. For function b, we use two features: one bias feature which is always 1 and s. Using these features, we can make an exact approximation with a weight vector of 1 and -0.5 respectively corresponding to the bias feature and s feature.

(b) yes: $(1, s, s^2)$

2. Consider the following neural network $\hat{v}$ with one-hidden layer, relu activation function g, with weights $\mathbf{W}^{[0]}, \mathbf{W}^{[1]}, \mathbf{b}^{[0]}, \mathbf{b}^{[1]}$,

$$\hat{v}(\mathbf{x}; \mathbf{W}^{[0]}, \mathbf{W}^{[1]}, \mathbf{b}^{[0]}, \mathbf{b}^{[1]}) = \mathbf{W}^{[1]} g(\mathbf{W}^{[0]}\mathbf{x} + \mathbf{b}^{[\mathbf{0}]}) + \mathbf{b}^{[1]}.$$

Recall that the following gradients

$$\begin{aligned}
\frac{\partial \hat{v}}{\partial \mathbf{W}^{[1]}_{ij}} &= g(\mathbf{W}^{[0]}\mathbf{x} + \mathbf{b}^{[\mathbf{0}]})_j \\
\frac{\partial \hat{v}}{\partial \mathbf{b}^{[1]}_{j}} &= 1 \\
\frac{\partial \hat{v}}{\partial \mathbf{b}^{[0]}_{i}} &= \sum_k \mathbf{W}^{[1]}_{ki} \frac{\partial g(\mathbf{W}^{[0]}\mathbf{x} + \mathbf{b}^{[0]})}{\partial \mathbf{b}^{[0]}_{i}} \\
\frac{\partial \hat{v}}{\partial \mathbf{W}^{[0]}_{ij}} &= \sum_k \mathbf{W}^{[1]}_{ki} \frac{\partial g(\mathbf{W}^{[0]}\mathbf{x} + \mathbf{b}^{[\mathbf{0}]})}{\partial \mathbf{W}^{[0]}_{ij}}
\end{aligned}$$

(a) What are the derivatives specifically for the relu activation g?

(b) We talked about carefully initializing the weights for the NN. For example, each weight can be sampled from a Gaussian distribution. Imagine instead you decided to initialize all the weights to zero. Why would this be a problem? Hint: Consider the derivatives in (a).

Answer:

(a)

$$\frac{\partial \hat{v}}{\partial \mathbf{W}^{[1]}_{ij}} = \max((\mathbf{W}^{[0]}\mathbf{x} + \mathbf{b}^{[\mathbf{0}]})_j, 0)$$

$$\frac{\partial \hat{v}}{\partial \mathbf{b}^{[1]}_{j}} = 1$$

$$\frac{\partial \hat{v}}{\partial \mathbf{b}^{[0]}_{i}} = \sum_k \mathbf{W}^{[1]}_{ki} 1_{(\mathbf{W}^{[0]}\mathbf{x}+\mathbf{b}^{[0]})>0}$$

$$\frac{\partial \hat{v}}{\partial \mathbf{W}^{[0]}_{ij}} = \sum_k \mathbf{W}^{[1]}_{ki} \mathbf{x}_j 1_{(\mathbf{W}^{[0]}\mathbf{x}+\mathbf{b}^{[0]})>0}$$

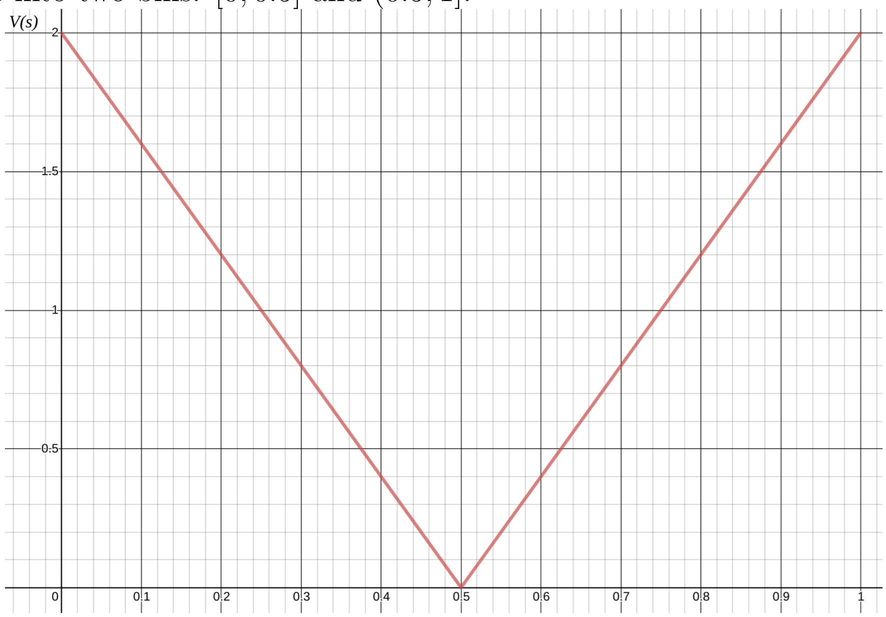
(b) If all the weights (including the biases) are zero, the derivative of the value function with respect to the weights will be zero as well. As a result, the updates will be zero and the weights do not get updated.

Worksheet 11

3. Consider a problem with the state space, $\mathcal{S} = \{0, 0.01, 0.02, \cdots, 1\}$. Assume the true value function is

$$v_\pi(s) = 4|s - 0.5|$$

which is visualized below. We decide to create features with state aggregation, and choose to aggregate into two bins: $[0, 0.5]$ and $(0.5, 1]$.



(a) What are the possible feature vectors for this state aggregation?

(b) Imagine you minimize the $\overline{\text{VE}}(\mathbf{w}) = \sum_{s \in \mathcal{S}} d(s)(v_\pi(s) - \hat{v}(s, \mathbf{w}))^2$ with a uniform weighting $d(s) = \frac{1}{101}$ for all $s \in \mathcal{S}$. What vector $\mathbf{w}$ is found?

(c) Now, if the agent puts all of the weighting on the range $[0, 0.25]$, (i.e. $d(s) = 0$ for all $s \in (0.25, 1]$), then what vector $\mathbf{w}$ is found by minimizing $\overline{\text{VE}}$?

Answer:

(a) $(0, 1)$ and $(1, 0)$

(b) With a uniform weighting, the weight vector will be $(1, 1)$

(c) we want to minimize the error only for states in $[0, 0.25]$. The weight vector will be $(1.5, 0)$. 1.5 is calculated by computing the value for $s = 0$ and $s = 0.25$: $v(0) = 2$ and $v(0.25) = 1$ and getting the average.

4. Consider the following general SGD update rule with a general target U_t

$$\mathbf{w}_{t+1} = \mathbf{w}_t + \alpha \left[U_t - \hat{v}(S_t, \mathbf{w}_t)\right] \nabla \hat{v}(S_t, \mathbf{w_t}).$$

Assume we are using linear function approximation, *i.e.* $\hat{v}(S, \mathbf{w}) = x(S)^\top \mathbf{w}$.

(a) What happens to the update if we scale the features by a constant and use the new features $\tilde{x}(S) = 2x(S)$? Why might this be a problem?

(b) In general we want a stepsize that is invariant to the magnitude of the feature vector $x(S)$, where the magnitude is measured by the inner product $x(S)^\top x(S)$. The book suggests the following stepsize:

$$\alpha = \frac{1}{\tau x(S)^\top x(S)}.$$

What is $x(S)^\top x(S)$ when using tile coding with 10 tilings? Suppose $\tau = 1000$, what is α if we use tile coding with 10 tilings?

Answer:

(a) Both $\hat{v}(S_t, \mathbf{w}_t)$ and $\nabla \hat{v}(S_t, \mathbf{w_t})$ become twice bigger. $\hat{v}(S_t, \mathbf{w}_t)\nabla \hat{v}(S_t, \mathbf{w_t})$ becoming 4 times bigger is as if the step-size is 4 times bigger. One problem that this could cause is that the step-size could become bigger than one. Therefore, we need to make the step-size 4 times smaller.

(b) $x(S)^\top x(S) = 10$ and $\alpha = 10^{-4}$