CS60050 Machine Learning - Weekly Report

Nisarg Upadhyaya (19CS30031)

Week 12: 3-5 November

1 Topics covered

- Critic and Learner
- K-arm Bandit
- Markov Decision Process
- Model Based Learning
- Value and Policy Iteration Algorithms
- Temporal Difference Algorithm

2 Summary

2.1 Critic and Learner

A critic evaluates the past performances of the learner. A learner interacts with the environment at some state and gets rewarded/penalized by the critic. Its target is to learn a series of actions to reach a goal state.

2.2 K-arm Bandit

In the deterministic version there are k leavers, where pulling a lever gives a certain reward, and we have to decide which leaver to pull to maximise the reward. We try different leavers and keep track of the best. Let a denote the action of pulling a certain lever and $Q(a)$ denote the reward earned. Initially $Q(a) = 0$ for all a. We then store the reward for each action and choose the action a^* if $Q(a^*) = max_a Q(a)$. In the non-deterministic case we have the action a as pulling a lever to win a reward r with probability $P(r|a)$. We keep $Q_t(a)$ as the estimate of the value of action a at time t. The updation takes place as $Q_{t+1}(a) \leftarrow Q_t(a) + \eta[r_{t+1}(a) - Q_t(a)]$ where η is the learning factor. Like before choose the action which maximises $Q(a)$.

We may have more complex environments with multiple states where actions affect the next state. The rewards are non-deterministic with a probability $P(r|s_i, a_j)$. We need to learn $Q(s_i, a_j)$ which is the value of taking action a_j in state s_i.

2.3 Markov Decision Process

An agent takes an action a_t at state s_t at a discrete time t. $P(r_{t+1}|s_t, a_t)$ and $P(a_{t+1}|s_t, a_t)$ are our model. A policy is a mapping $\pi : S \rightarrow A$. The discounted value with infinite horizon is given as $V^\pi(s_t) = E[r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + ...]$. Optimal policy is one which maximises this value. Alternatively we can also learn $Q(s_t, a_t)$. So the optimal policy for state s_t becomes choosing an action a^* which maximises $V^\pi(s_t)$ or $Q(s_t, a_t)$.

2.4 Model Based Learning

$P(r_{t+1}|s_t, a_t)$ and $P(a_{t+1}|s_t, a_t)$ are known and we directly solve for the optimal value function and policy using dynamic programming. There are two broad approaches.

2.4.1 Value iteration algorithm

Initialise $V(s)$ to arbitrary values. Now until $V(s)$ converges repeat for all $s \in S$ and all $a \in A$ set $Q(s, a) \leftarrow E[r|s, a] + \gamma \sum_{s' \in S} P(s'|s, a)V(s')$ and then set $V(s) \leftarrow max_a Q(s, a)$.

2.4.2 Policy iteration algorithm

Initialise a policy π' arbitrarily. Repeat the following steps until $\pi = \pi'$. Set $\pi \leftarrow \pi'$. Compute the values using π by solving $V^\pi(s) = E[r|s, \pi(s)] + \gamma \sum_{s' \in S} P(s'|s, \pi(s))V^\pi(s')$. Improve the policy at each state by setting $\pi'(s) \leftarrow argmax_a(E[r|s, a] + \gamma \sum_{s' \in S} P(s'|s, a)V^\pi(s'))$.

2.5 Temporal Difference Algorithm

We have no prior knowledge of the model, i.e., $P(r_{t+1}|s_t, a_t)$ and $P(a_{t+1}|s_t, a_t)$ are not known. The environment needs to be explored to query the model and see the value of the next state and reward. These algorithms examine the difference between the current estimate of the value and the discounted value of the next state.

2.5.1 Deterministic environment

For any state action (s_t, a_t) a single reward r_{t+1} and transition s_{t+1} is possible.

$$Q(s_t, a_t) = r_{t+1} + \gamma max_{a_{t+1}} Q(s_{t+1}, a_{t+1})$$

We update this at every exploration and converge when little change is observed between consecutive iterations.

2.5.2 Non-deterministic environment

The reward and next state for a state action pair are not fixed. We keep a running average of values and updates are as follows. This is the *Q learning* algorithm.

$$\hat{Q}(s_t, a_t) \leftarrow \hat{Q}(s_t, a_t) + \eta(r_{t+1} + \gamma max_{a_{t+1}} \hat{Q}(s_{t+1}, a_{t+1}) - Q(s_t, a_t))$$

We choose the next action randomly. Initially sample an action with a probability ϵ. Later actions providing higher values would have higher probabilities. This is called ϵ-Greedy sampling.

2.5.3 Issues

Note that these techniques are not feasible through tabular search due to large number of states and actions. We use regression to predict Q values given current value, reward and next state.

3 Challenging concepts

Temporal difference algorithms

4 Interesting concepts

None

5 Concepts not understood

None

6 Ideas

We have previously looked at the importance of model/learner selection in ensemble learning. Taking inspiration from the K-arm bandit problem we can have a RL model which helps us select an optimal learner for a specific ML task from a group of trained learners where while training the reward is the accuracy the learner gives on the test dataset.