ECE433/COS435 Introduction to RL
Assignment 0: Review of Topics
Spring 2024

Fill me in

catherine ji

Due February 4, 2024

Collaborators

Fill me in

none, side note: I am on the waitlist for this class

Instructions

You should work alone for this assignment. Writeups should be typeset in Latex and submitted as PDF. You can work with whatever tool you like for the code, but **please submit the asked-for snippet and answer in the solutions box as part of your writeup. We will only be grading your writeup.** Make sure to still also attach your notebook/code with your submission.

Introduction

This Homework is meant to be a review of standard topics on machine learning, linear algebra, and probability, with a heavy emphasis on topics that will reappear in later stages of this course.

You will also work with PyTorch and Python to build your models, so we want to make sure you are familiar with the specific packages (e.g. `torch.distributions`) that you will be using throughout the course.

Question 1. Probability

In a reinforcement learning setting, we are often interested in settings where an agent interacts with a simple game. We often want to learn the agent's policy π, or its "rule" for making actions. In this problem, you will derive properties of different "rules" over a simple game, more interesting parts of the course.

Suppose we are playing a really simple game with 10 unique boxes, each with their own distinct label from 1, ..., 10. One of these boxes contains a golden coin that awards the player with a reward of 5, one contains a silver coin that awards the player with a reward of 1, and the rest contain nothing and award the player with a reward of 0 if selected. The player must open a single box, and based on the outcome, their final score is the reward corresponding to the box that they selected. More formally, the player has a policy π and uses it to select an action $a \sim \pi$ corresponding to the box the player picked (e.g. $a = 1$ means the player picked box 1). The player is then given a reward $r(a)$ based on their action.

Question 1.a

Suppose our **agent's policy** is to randomly select one of the labelled boxes, each with equal probability. In other words, $\pi = \mathcal{U}\{1, \ldots, 10\}$, and the agent selects a box $a \sim \pi$. Conditioned on the fact the the gold coin is in the box labelled 5 and the silver coin is in box labeled 1, what is the expected reward?

Solution

Let R be the random variable equal to the reward from the random selection and r_i be the reward from opening box i, where $r_5 = 5$, $r_1 = 1$, and $r_j = 0$ for $j \neq 1, 5$. By definition of expectation value, $E[R] = \sum_{i=1}^{10} p_i r_i = \frac{1}{2} + \frac{1}{10} = \frac{3}{5}$.

Question 1.b

Given the same assumptions as (1.a), what is the policy of the agent that maximizes the expected reward?

Solution

The optimal policy is $\pi = (0 \text{ if box} \neq 5, 1 \text{ if box} = 5) = \delta_{i,5}$. The expected reward $E[R] = \sum_{i=1}^{10} p_i r_i = 5$, which is clearly maximized because $\max_i r_i = 5$ and $E[R] \leq \max_i r_i = 5$.

Question 1.c

What is the entropy of the policy in (1.a) and the entropy of the policy in (1.b)?

Solution

Entropy S is defined here as $S = -\sum_{i=1}^{10} p_i \log p_i$. Using this definition, for 1.a, the policy has entropy $S = \log 10$. For 1.b, the policy has entropy $S = 0$.

Question 1.d Coding

Suppose instead of boxes, we now deal with a *continuous* set of possible decisions. In this scenario, our agent can choose any number $a \in \mathbb{R}$, and the reward given to the player is defined by

$$\mathcal{R}(x) = \max\left\{0, 1 - |x - 1|\right\}$$

In this problem, you will use this reward function to compute the expected return of a policy (to be defined). The key idea is that an expectation can be approximated through sampling (recall Monte Carlo sampling from COS324). In this problem, you will familiarize yourself with the `torch.distributions` library.

See the colab notebook (1.d) for more details. Fill out the indicated code segments below after solving the problem.

Solution

Expected reward is **0.58 to the nearest hundredth, 0.6 to the nearest tenth as specified in the ipynb file**.

Fill in your code here:

```
trials = 10000
distribution = Normal(0.6,0.3)
results = distribution.sample((1,trials))

r = reward(results)
print("Expected reward:", r.mean())
```

Question 1.e Coding

Plot the reward distribution under this policy using `matplotlib.pyplot.hist()` or some other histogram plotting library. Make sure to label your axes for this problem (x-axis is reward, y-axis is probability density). See colab notebook (1.e) for more details.

Hint. *When using `matplotlib.pyplot.hist()`, remember it needs to be a PDF! (check the documentation.)*

Solution

```python
plt.hist(r,30,density=True)
plt.xlabel("reward")
plt.ylabel("probability density")
plt.xlim(0,1)
plt.show()
```

1e_plot.png

Question 2. MLE

The purpose of this question is to review Maximum Likelihood Estimation (MLE), which appears in many machine learning settings. In previous courses, students may have seen or solved a few questions on computing the Maximum Likelihood Estimator for a set of datapoints, in which case this problem will be review.

A random variable X has a Normal distribution with parameters μ, σ^2 if its probability distribution is uniquely defined as

$$f_X(x) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$$

Question 2.a

Let $\theta = \sigma^2$. For a set of datapoints $X_1, ..., X_n$ sampled i.i.d. from a Normal distribution with unknown parameters μ and θ, find the log-likelihood function $L(\mu, \theta; X_1, ..., X_n)$. You answer should be in terms of n, $X_1, ..., X_n$, μ, and θ.

Solution

By definition, $L(\mu, \theta; \mathbf{X}) = P(\mathbf{X}|\mu, \theta) = \Pi_{i=1}^n P(X_i|\mu, \theta) = \Pi_{i=1}^n f_X(X_i; \mu, \theta)$ since X_i are independent. Taking the log,

$$\begin{aligned}\log L(\mu, \theta; X_1, ..., X_n) &= \sum_{i=1}^n \log\Big(\frac{1}{\sqrt{2\pi\theta}} \exp\Big(-\frac{(X_i-\mu)^2}{2\theta}\Big)\Big) \\ &= -\Big(\sum_{i=1}^n \frac{(X_i-\mu)^2}{2\theta}\Big) - n\log\sqrt{2\pi\theta}.\end{aligned}$$

Question 2.b

Using your answer in (2.a), solve for the MLE of μ and θ. Solutions should also show that the critical points are maximums e.g. compute derivatives around the critical point and number of critical points argument.

Solution

Critical points are defined as $(\hat{\mu}, \hat{\theta})$ where $\frac{\partial L}{\partial \mu}|_{(\hat{\mu},\hat{\theta})} = 0$ and $\frac{\partial L}{\partial \hat{\theta}}|_{(\hat{\mu},\hat{\theta})} = 0$. Taking the gradient, where $\bar{X} = \frac{1}{n}\sum_{i=1}^n X_i$,

$$\frac{\partial L}{\partial \mu}|_{(\hat{\mu},\hat{\theta})} = \sum_{i=1}^n \frac{X_i - \hat{\mu}}{\hat{\theta}} = \frac{n(\bar{X}-\hat{\mu})}{\theta} = 0$$

$$\frac{\partial L}{\partial \theta}|_{(\hat{\mu},\hat{\theta})} = \sum_{i=1}^n -\frac{1}{2\hat{\theta}} + \frac{(X_i-\hat{\mu})^2}{2\hat{\theta}^2} = 0 \rightarrow \sum_{i=1}^n -\hat{\theta} + (X_i - \hat{\mu})^2 = 0$$

Solving for $\hat{\mu}$ and $\hat{\theta}$,

$$\hat{\mu} = \bar{X} = \frac{1}{n}\sum_{i=1}^{n} X_n$$

$$\hat{\theta} = \frac{1}{n}\sum_{i=1}^{n}(X_i - \hat{\mu})^2$$

.

To prove that $(\hat{\mu}, \hat{\theta})$ is a global maximum, we note that $\text{sign}(\frac{\partial L}{\partial \mu}|_{(\hat{\mu}+\epsilon_\mu, \hat{\theta}+\epsilon_\theta)}) = -\text{sign}(\epsilon_\mu)$ and $\text{sign}(\frac{\partial L}{\partial \theta}|_{(\hat{\mu}+\epsilon_\mu, \hat{\theta}+\epsilon_\theta)}) = -\text{sign}(\epsilon_\theta)$ in the vicinity of the critical point which implies that $(\hat{\mu}, \hat{\theta})$ is a local maximum. In addition, there is exactly one critical point in (μ, θ) space, so the local maximum must also be a global maximum, so we are done. Alternatively, one can show the hessian of L w.r.t. θ and μ is negative definite. ■

Question 3. Coding

The objective of this question is to familiarize you with the basic mechanics of PyTorch and Python. See **Question 3** on the .ipynb notebook, but you will be coding up gradient descent and building a basic neural network model.

Question 3.a

For this problem and (3.b), see the provided .ipynb notebook for all the questions, and fill out the indicated code segments below after solving each problem.

Solution

```
# Import the dataset
rng = np.random.RandomState(189289213)
N = 1000
X = 10 * rng.rand(N, 10) # feature matrix
y = np.dot(X, [1,2,3,4,5,6,7,8,9,10]) + np.random.normal(0, 0.01) #
    target vector
y_col = np.array([y]).T # target vector as a column vector

def init_weights (in_1:int, in_2:int) -> np.ndarray:
  return np.random.rand(in_1,in_2)

weights = init_weights(X.shape[1], 1)

# Helper functions to get you started. Feel free to use or not use
    them.
def gradient_descent(X, y, weights: np.ndarray, eta: float,
    iterations: int):
  for i in range(iterations):
    weights -= eta * grad(X, y, weights).T

  return weights

def grad(X, y, weights: np.ndarray): # numerator picture
  temp_grad = (1/(2*N)) * 2 * (X@weights - y_col).T @ X
  return temp_grad

resultWeights = gradient_descent(X,y,weights,0.001,iterations=10000)
print("Weights after gradient descent: \n", resultWeights)
```

Question 3.b

Rewrite your solution to (3.a) using the torch library. Your solution must use `loss.backward()` and `weight.grad`.

Solution

```python
X_t = torch.from_numpy(X).float()
y_t = torch.from_numpy(y_col).float() # so y_t is a column vector

weights = torch.randn((X.shape[1], 1), requires_grad=True)
loss_fxn = torch.nn.MSELoss()

def gradient_descent_torch(X, y, weights: torch.tensor, eta: float,
    iterations: int) -> None:
  for i in range(iterations):
    loss = loss_fxn(torch.matmul(X_t,weights),y_t)
    loss.backward()
    with torch.no_grad(): # so grad call doesn't keep track of
    updates
      weights -= eta * weights.grad
    weights.grad.zero_() # or else weights.grad will call the
    cumulative sum of gradients; is useful for RNN

  return weights

resultWeights = gradient_descent_torch(X_t, y_t, weights, 0.001,
    iterations=1000)
print("Weights after gradient descent: \n", resultWeights)
```

Question 4. Feedback

We value your feedback and would love to hear your thoughts on the course. Please share your feedbacks, including what you expect to change in course structure or logistics, areas for improvement, and any specific expectations or topics you would like us to cover in the future. Additionally, we welcome any suggestions on how we can enhance the learning experience to better meet your needs. Your input is greatly appreciated!

Solution

Your solution here.