ECE59500RL Homework 5

Robert (Cars) Chandler — chandl71@purdue.edu

Problem 1

1.1

The objective function for ERM with square loss is:

$$\min_{f \in \mathcal{F}} \sum_{i=1}^{N} \|\hat{y} - y\|^2$$

And in this case, $\hat{y} = Q_\theta$ which comes from a class of functions parameterized by the four θ values:

$$\min_{f \in \mathcal{F}} \sum_{i=1}^{N} \| \Big(\theta_1 s_i^2 + \theta_2 a_i^2 + \theta_3 s_i a_i + \theta_4 \Big) - y \|^2$$

So we want to find the $Q \in \mathcal{Q}$ that yields the least squared error, which is equivalent to finding $\theta \in \mathbb{R}^4$, the parameterization of $Q \in \mathcal{Q}$ yielding the least squared error:

$$\hat{Q}^{\pi_t} = \arg\min_{\theta \in \mathbb{R}^4} \sum_{i=1}^{N} \| \Big(\theta_1 s_i^2 + \theta_2 a_i^2 + \theta_3 s_i a_i + \theta_4 \Big) - y \|^2$$

We are given three iterations of data ($N = 3$) which we can substitute into this form to give the final result of the objective function:

$$\arg\min_{\theta\in\mathbb{R}^4} \sum_{i=1}^{N} \| \Big(\theta_1 s_i^2 + \theta_2 a_i^2 + \theta_3 s_i a_i + \theta_4\Big) - y\|^2$$

$$\arg\min_{\theta\in\mathbb{R}^4} \Big[\Big(\theta_1 \cdot 1^2 + \theta_2 \cdot 0^2 + \theta_3 \cdot 1 \cdot 0 + \theta_4 - 1\Big)^2 + \Big(\theta_1 \cdot (-2)^2 + \theta_2 \cdot 1^2 + \theta_3 \cdot -2 \cdot 1 + \theta_4 - 0\Big)^2 + \Big(\theta_1 \cdot 1^2 + \theta_2 \cdot (-1)^2 + \theta_3 \cdot 1 \cdot -1 + \theta_4 - 3\Big)^2\Big]$$

$$\arg\min_{\theta\in\mathbb{R}^4} \Big[(\theta_1 + \theta_4 - 1)^2 + (4\theta_1 + \theta_2 - 2\theta_3 + \theta_4)^2 + (\theta_1 + \theta_2 - \theta_3 + \theta_4 - 3)^2\Big]$$

1.2

$$\pi_{t+1}(a|s) = (1-\alpha)\pi_t(a|s) + \alpha\bar{\pi}(a|s)$$

We know that π_0 is a uniform distribution across the action space:

$$\pi_0(a|s) = \left[\frac{1}{3}, \frac{1}{3}, \frac{1}{3}\right], \quad \forall s \in \mathcal{S}$$

Next, we need to calculate $\bar{\pi}$ according to:

$$\bar{\pi}(s) = \arg\max_{a\in\mathcal{A}} \hat{Q}^{\pi_t}(s,a), \forall s \in \mathcal{S}$$

We can iterate over combinations of the relevant subset of the state-action space to determine which action maximizes the quantity above and then calculate the updated policy according to the definition above:

```
import itertools

import numpy as np
import xarray as xr
from IPython.display import Markdown

state_space = np.arange(-3, 4)
action_space = np.arange(-1, 2)


def qhat0(s, a):
    return 2 * s**2 + a**2 - s * a + 0.5
```

```python
# Initialize qhat0
q_eval = xr.DataArray(
    data=np.zeros([len(state_space), len(action_space)]),
    coords={"s": state_space, "a": action_space},
)

# Evaluate qhat0 at all (s, a)
for s, a in itertools.product(state_space, action_space):
    q_eval.loc[dict(s=s, a=a)] = qhat0(s, a)

# Get the optimal actions for qhat0
i_optimal_action = q_eval.argmax(dim="a").to_numpy()  # type: ignore

optimal_actions = action_space[i_optimal_action]

# Uniform distribution
p0 = xr.full_like(q_eval, 1 / 3)

pbar = xr.zeros_like(q_eval)

# We calculated optimal actions, so we "pick" the action accordingly (assign it
# probability 1)
for s, a in zip(state_space, optimal_actions):
    pbar.loc[dict(s=s, a=a)] = 1

alpha = 0.25

# Convex combination of policies
p1 = (1 - alpha) * p0 + alpha * pbar

# Get the distributions for s = 1, 2
p1_s1 = Markdown(str(p1.sel(s=1).data.tolist()))
p1_s2 = Markdown(str(p1.sel(s=2).data.tolist()))
```

So the probability distribution of action values (in the same order they appear in the action space definition) are:

$$\pi_1(a|1) = [0.5, 0.25, 0.25]$$
$$\pi_1(a|2) = [0.5, 0.25, 0.25]$$

Problem 2

2.1

For $s \in \mathcal{S} \in \tau$:

$$a^* = \arg\max_{a \in \mathcal{A}} \hat{Q}(s, a)$$

$$\begin{aligned} a^*(s = b) &= x \\ a^*(s = c) &= x \\ a^*(s = d) &= x \\ a^*(s = e) &= y \end{aligned}$$

For $a \in \mathcal{A}$:

$$\pi(a|s) = 1 - \varepsilon + \frac{\varepsilon}{|\mathcal{A}|} \text{ if } a = a^* \text{ else } \frac{\varepsilon}{|\mathcal{A}|}$$

$$\begin{aligned} \pi(x|b) &= 1 - 0.1 + \frac{0.1}{2} = 0.95 \quad & \pi(y|b) &= \frac{0.1}{2} = 0.05 \\ \pi(x|c) &= 1 - 0.1 + \frac{0.1}{2} = 0.95 \quad & \pi(y|c) &= \frac{0.1}{2} = 0.05 \\ \pi(x|d) &= 1 - 0.1 + \frac{0.1}{2} = 0.95 \quad & \pi(y|d) &= \frac{0.1}{2} = 0.05 \\ \pi(x|d) &= \frac{0.1}{2} = 0.05 & \pi(y|e) &= 1 - 0.1 + \frac{0.1}{2} = 0.95 \end{aligned}$$

2.2

The dataset of three points is formed by three (x_i, y_i) pairs where $x_i = (s_i, a_i)$ and

$$y_i = r_i + \gamma \max_{a' \in \mathcal{A}} \hat{Q}(s_{i+1}, a')$$

The x_i are trivial to form, and we can calculate the y_i here:

$$
\begin{aligned}
y_t &= r_t + 0.9 \max_{a' \in \mathcal{A}} \hat{Q}(s_{t+1}, a') \\
&= 1 + 0.9 \cdot 0.75 \\
&= 1.675
\end{aligned}
$$

$$
\begin{aligned}
y_{t+1} &= r_{t+1} + 0.9 \max_{a' \in \mathcal{A}} \hat{Q}(s_{t+2}, a') \\
&= -2 + 0.9 \cdot -1.5 \\
&= -3.35
\end{aligned}
$$

$$
\begin{aligned}
y_{t+2} &= r_{t+2} + 0.9 \max_{a' \in \mathcal{A}} \hat{Q}(s_{t+3}, a') \\
&= -0.5 + 0.9 \cdot -1.5 \\
&= -1.85
\end{aligned}
$$

So the dataset is:

$$\{(x_t = (c, x), y_t = 1.675), (x_{t+1} = (e, x), y_{t+1} = -3.35), (x_{t+2} = (b, y), y_{t+2} = -1.85)\}$$

2.3

The objective function for ERM with square loss is:

$$\min_{f \in \mathcal{F}} \sum_{i=1}^{N} \|\hat{\mathbf{y}} - \mathbf{y}\|^2$$

FIXME: do we want to add an argmin form here as well?

where $\hat{\mathbf{y}} = \boldsymbol{\theta}^\top \boldsymbol{\phi}(s, a)$. We can calculate $\hat{y}$ for each data point:

For $x_t = (c, x)$:

$$\hat{y}_t = \begin{bmatrix} \theta_1 & \theta_2 \end{bmatrix} \begin{bmatrix} -1 \\ 1 \end{bmatrix} = \theta_2 - \theta_1$$

For $x_{t+1} = (e, x)$:

$$\hat{y}_t = \begin{bmatrix} \theta_1 & \theta_2 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \theta_2$$

For $x_{t+2} = (b, y)$:

$$\hat{y}_t = \begin{bmatrix} \theta_1 & \theta_2 \end{bmatrix} \begin{bmatrix} -2 \\ -1 \end{bmatrix} = -2\theta_1 - \theta_2$$

So using these with our values for y from the dataset, the objective function becomes:

$$\min_{\boldsymbol{\theta} \in \mathbb{R}^2} \sum_{i=1}^{N} \|\hat{\mathbf{y}}_i - \mathbf{y}_i\|^2$$
$$= \min_{\boldsymbol{\theta} \in \mathbb{R}^2} \left[(\theta_2 - \theta_1 - 1.675)^2 + (\theta_2 + 3.35)^2 + (-2\theta_1 - \theta_2 + 1.85)^2 \right]$$

2.4

We can solve this by forming a simple design matrix from the feature vectors we have and then finding the least squares solution using that with the output vector we have from the y_i values:

```
import numpy as np
from IPython.display import Markdown

a = np.array([[-1, 1], [0, 1], [-2, -1]])
b = np.array([1.675, -3.35, -1.85])
theta, *_ = np.linalg.lstsq(a, b)

t1, t2 = [Markdown(f"{theta[i]:.5f}") for i in [0, 1]]
```

$$\theta_1 = 0.42143$$
$$\theta_2 = -0.08214$$

Problem 4

4.1

$$\mathcal{L}(\pi, \beta) = \mathbb{E}_{A \sim \pi(\cdot|s)} \left[Q^{\pi_t}(s, A) \right] + \lambda \mathcal{H}(\pi(\cdot|s)) - \beta \left(\sum_{a \in \mathcal{A}} \pi(a|s) - 1 \right)$$

We expand each term into its respective summation and differentiate:

$$\frac{\partial \mathcal{L}}{\partial \pi(a|s)} = \frac{\partial}{\partial \pi(a|s)} \left[\sum_{a' \in \mathcal{A}} \pi(a'|s) Q^{\pi_t}(s, a') - \lambda \sum_{a' \in \mathcal{A}} \pi(a'|s) \log \pi(a'|s) - \beta \left(\sum_{a' \in \mathcal{A}} \pi(a'|s) - 1 \right) \right]$$

$$\frac{\partial \mathcal{L}}{\partial \pi(a|s)} = Q^{\pi_t}(s, a) - \lambda \left(1 + \log \pi(a|s)\right) - \beta, \quad \forall a \in \mathcal{A}$$

The differentiation can be explained as follows: the first term is a summation of policy terms across each action, and the partial derivative will only evaluate to a nonzero value when $a' = a$, at which point it differentiates to one, so we end up just getting the constant Q from that summation. Similarly, the second summation only evaluates when $a' = a$, but this time we use the product rule to evaluate:

$$\frac{\partial}{\partial \pi(a|s)} \pi(a|s) \log \pi(a|s) = \frac{\pi(a|s)}{\pi(a|s)} + \log \pi(a|s) \cdot 1 = 1 + \log \pi(a|s)$$

Lastly, the final summation evaluates to 1 when $a' = a$ and at this point, we just get the constant β as a result of our differentiation since the subtracted 1 is a constant eliminated by the differentiation.

4.2

The derivative with respect to β is much simpler as every term except the last summation cancels out:

$$\frac{\partial \mathcal{L}}{\partial \beta} = \frac{\partial}{\partial \beta} \left[\sum_{a' \in \mathcal{A}} \pi(a'|s) Q^{\pi_t}(s, a') + \lambda \sum_{a' \in \mathcal{A}} \pi(a'|s) \log \pi(a'|s) - \beta \left(\sum_{a' \in \mathcal{A}} \pi(a'|s) - 1 \right) \right]$$

$$\frac{\partial \mathcal{L}}{\partial \pi(a|s)} = 1 - \sum_{a' \in \mathcal{A}} \pi(a'|s)$$

4.3

Beginning with $\frac{\partial \mathcal{L}}{\partial \pi(a|s)} = 0$:

$$\begin{aligned}
0 &= Q^{\pi_t}(s, a) - \lambda \left(1 + \log \pi(a|s)\right) - \beta \\
\lambda \left(1 + \log \pi(a|s)\right) &= Q^{\pi_t}(s, a) - \beta \\
\log \pi(a|s) &= \frac{Q^{\pi_t}(s, a) - \beta}{\lambda} - 1 \\
\pi(a|s) &= \exp\left(\frac{Q^{\pi_t}(s, a) - \beta}{\lambda} - 1 \right)
\end{aligned}$$

We now look at $\frac{\partial \mathcal{L}}{\partial \beta} = 0$:

$$0 = 1 - \sum_{a' \in \mathcal{A}} \pi(a'|s)$$
$$\sum_{a' \in \mathcal{A}} \pi(a'|s) = 1$$

We can use this to continue solving for π where we previously left off:

$$\sum_{a' \in \mathcal{A}} \pi(a'|s) = \sum_{a' \in \mathcal{A}} \exp\left(\frac{Q^{\pi_t}(s,a) - \beta}{\lambda} - 1\right)$$
$$1 = \sum_{a' \in \mathcal{A}} \exp\left(\frac{Q^{\pi_t}(s,a) - \beta}{\lambda}\right) e^{-1}$$
$$e = \sum_{a' \in \mathcal{A}} \exp\left(\frac{Q^{\pi_t}(s,a) - \beta}{\lambda}\right)$$
$$e = \sum_{a' \in \mathcal{A}} \exp\left(\frac{Q^{\pi_t}(s,a)}{\lambda}\right) e^{-\beta/\lambda}$$
$$\exp\left(\frac{\beta}{\lambda} + 1\right) = \sum_{a' \in \mathcal{A}} \exp\left(\frac{Q^{\pi_t}(s,a)}{\lambda}\right)$$

Now we can substitute this back into our equation for $\pi(a|s)$:

$$\pi(a|s) = \exp\left(\frac{Q^{\pi_t}(s,a)}{\lambda}\right) \exp\left(-\frac{\beta}{\lambda} - 1\right)$$
$$\pi(a|s) = \frac{\exp\left(\frac{Q^{\pi_t}(s,a)}{\lambda}\right)}{\sum_{a' \in \mathcal{A}} \exp\left(\frac{Q^{\pi_t}(s,a')}{\lambda}\right)}$$

This is the final solution for the stationary point of the Lagrangian. It is the softmax policy function that we saw in Problem 2.

4.4

TODO

Problem 5

5.1

We can represent the reward function and trajectories as data structures and compute the discounted return for each sub-tracectory in Python:

```
import numpy as np
import xarray as xr
from IPython.display import Markdown

reward = xr.DataArray(
    data=np.array(
        [
            [-1, 1, 0, 0, 0],
            [-2, 0, 0, 3, -5],
            [-2, -2, 5, 0, 0],
            [-2, -2, 4, 0, 0],
        ]
    ),
    coords=dict(
        state=["empty", "monster", "food", "resource"],
        action=["stay", "explore", "collect", "evade", "befriend"],
    ),
)

trajectories = [
    [("empty", "stay"), ("monster", "evade"), ("resource", "collect")],
    [("empty", "explore"), ("food", "collect"), ("monster", "befriend")],
    [("empty", "explore"), ("empty", "explore"), ("resource", "collect")],
]

gamma = 0.95


returns = np.sum(
    np.array(
        [
            [
                gamma**t * reward.sel(state=s, action=a).item()
                for t, (s, a) in enumerate(traj)
            ]
```

```
            for traj in trajectories
        ]
    ),
    axis=1,
)

r1, r2, r3 = returns

return_latex = Markdown(
    "\n".join([rf"\tau_{i+1}&: {r:0.4f} \\" for i, r in enumerate(returns)])
)

pref_1_over_2 = np.exp(r1) / np.sum([np.exp(r1), np.exp(r2)])
pref_1_over_3 = np.exp(r1) / np.sum([np.exp(r1), np.exp(r3)])
```

Note: we also calculate the human preference probabilities at the end of this code, used in the next two parts.

The discounted returns for each sub-trajectory are as follows:

$$\tau_1 : 5.4600$$
$$\tau_2 : 1.2375$$
$$\tau_3 : 5.5600$$

5.2

The values are calculated at the end of the code block in 5.1 according to the following equation:

$$\mathbb{P}[a_1 > a_2|s] = \frac{\exp\left(r^*(s, a_1)\right)}{\exp\left(r^*(s, a_1)\right) + \exp\left(r^*(s, a_2)\right)}$$

According to $\hat{R}$, the probability the human chooses τ_1 over τ_2 is:

$$\mathbb{P}[\tau_1 > \tau_2|s]$$