CMPSC 448: Machine Learning and Algorithmic AI
Homework 5

Instruction

This HW includes both theory and implementation problems. Please note,

- Your code must work with Python 3.7+
- You need to submit a report in PDF including all written deliverable, and all implementation codes so one could regenerate your results.
- For programming part of PCA problem, you can import any module you would like from scikit-learn or other external libraries to minimize the amount of implementation required to get the coding problem done. Submit your code in Problem.py.

PCA

Problem 1. [30 points] In this assignment, you will explore PCA as a technique for discerning whether low-dimensional structure exists in a set of data and for finding good representations of the data in that subspace. To this end, you will do PCA on a Iris dataset which can be loaded in scikit-learn using following commands:

```
from sklearn.datasets import load_iris iris = load_iris()
X = iris.data
y = iris.target
```

1. Carry out a principal component analysis on the entire **raw** dataset (4-dimensional instances) for $k = 1, 2, 3, 4$ components. How much of variance in data is retained when $k = 1, 2, 3, 4$, respectively?
2. Apply the centering and standardization operations from Problem 1 on raw data set and repeat part (1) on processed data. Explain any differences you observe compared to part (1) and justify.
3. Project the raw four dimensional data down to a two dimensional subspace generated by first two top principle components (PCs) from part (2) and produce a scatter plot of the data. Make sure you plot each of the three classes differently (using color or different markers). Can you see the three Iris flower clusters?
4. Either use your k-means++ implementation from previous homework or from scikit-learn to cluster data from part (3) into three clusters. Explain your observations.

Matrix Factorization

Problem 2. [30 points] Recall the following objective for extracting latent features from a partially observed rating matrix via matrix factorization (MF) for making recommendations, discussed in the class:

$$\min_{\boldsymbol{U},\boldsymbol{V}} \left[f(\boldsymbol{U},\boldsymbol{V}) \equiv \sum_{(i,j)\in\Omega} (r_{i,j} - \boldsymbol{u}_i^\top \boldsymbol{v}_j)^2 + \alpha \sum_{i=1}^{n} \|\boldsymbol{u}_i\|_2^2 + \beta \sum_{j=1}^{m} \|\boldsymbol{v}_j\|_2^2 \right] \tag{1}$$

where

- n: number of users
- m: number of items
- $r_{i,j}$: (i,j)-th element in $\boldsymbol{R} \in \mathbb{R}^{n\times m}$ input partially observed rating matrix
- $\Omega \subseteq [n] \times [m]$: index of observed entries in rating matrix, where $[n]$ denotes the sequence of numbers $\{1, 2, \ldots, n\}$.
- k: number of latent features
- $\boldsymbol{U} \in \mathbb{R}^{n\times k}$: the (unknown) matrix of latent feature vectors for n users (the ith row $\boldsymbol{u}_i \in \mathbb{R}^k$ is the latent features for ith user)
- $\boldsymbol{V} \in \mathbb{R}^{m\times k}$: the (unknown) matrix of latent feature vectors for m items (the jth row $\boldsymbol{v}_j \in \mathbb{R}^k$ is the latent features for jth movie)

Please do the followings:

1. In solving Equation 1 with iterative Alternating Minimization algorithm (fixing $\boldsymbol{V}^{(t)}$ and taking gradient step for $\boldsymbol{U}^{(t)}$ and vice versa), discuss what happens if $\boldsymbol{U}^{(0)}$ and $\boldsymbol{V}^{(0)}$ are initialized to zero?
2. Discuss why when there is no regularization in basic MF formulated in Equation 1, i.e., $\alpha = \beta = 0$, each user must have rated at least k movies, and each movie must have been rated by at least k users (Hint: consider the closed form solution for $\boldsymbol{u}_i$ and $\boldsymbol{v}_j$ in notes and argue why these conditions are necessary without regularization).
3. Computing the closed form solution in part (2) could be computational burden for large number of users or movies. A remedy for this would be using iterative optimization algorithms such as Stochastic Gradient Descent (SGD) where you sample movies in updating the latent features for users and sample users in updating the latent features of movies. Derive the updating rules for $\boldsymbol{u}_i^{(t)}$ and $\boldsymbol{v}_j^{(t)}$ at tth iteration of SGD. Show the detailed steps.

MDP and RL

Problem 3. [40 points] Consider the MDP in Figure 1. There are five states {A, B, C, D, T} and two actions $\{a_1, a_2\}$. The state T is the terminal state. The numbers on each arrow show the probability of moving to the next state and the reward, respectively. For example, if the agent takes action a_1 at state A, with probability 0.7 it will move to state B and will be rewarded -10, and with probability 0.3 it will move to state C and will be rewarded -5.

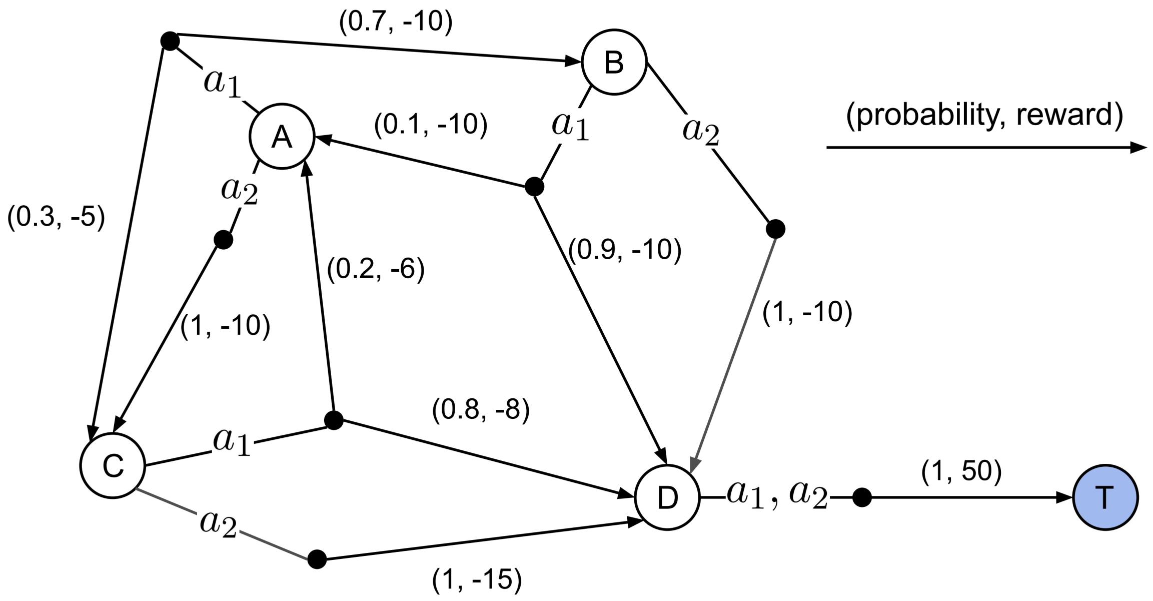

Figure 1: MDP

(1) [10 points] For a policy π that always takes action a_1 at every state, write down the Bellman Equation for state value function for each state using discount factor γ.

(2) [10 points] From the Bellman Equation in (1), compute the state values using $\underline{\gamma = 0.5}$.

(3) [5 points] Consider a random policy π_0 that uniformly selects actions at each state (i.e., the probability of taking each of two actions under this policy is 0.5). Initialize $v_0(s) = 0$ for all states, apply one sweep of policy evaluation to show $v_1(s)$ for all states using $\underline{\gamma = 1}$.

(4) [5 points] Based on the result of (3), apply policy improvement by greedification with $\underline{\gamma = 1}$ to show the improved policy π_1. Select actions uniformly if there is a tie.

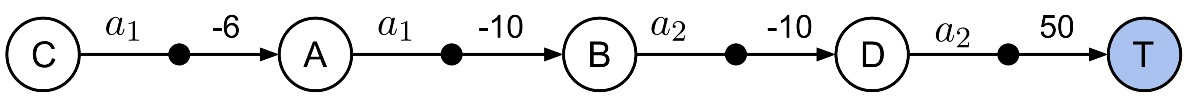

Figure 2: An Episode for MDP in Figure 1.

Now let's do Temporal Difference Control on this MDP using SARSA and Q-Learning. We initialize $Q(s, a) = 0$ for all pairs of (s, a), and we use step size $\underline{\alpha = 0.1}$ and discount factor $\underline{\gamma = 0.5}$. Suppose we sample an episode in Figure 2.

(5) [5 points] Use SARSA to update Q values by processing the episode from beginning to end.

(6) [5 points] Use Q-Learning to update Q values by processing the episode from beginning to end.

Bonus

Problem 4. [20 points] As we discussed in the lecture, the PCA algorithm is sensitive to scaling and pre-processing of data. In this problem, we explore few data pre-processing operations on data matrices.

1. Explain what does each of the following data processing operation mean, why it is important, and how you apply to a data matrix $\boldsymbol{X} \in \mathbb{R}^{n \times d}$, n samples each with d features [1] (if it helps, use mathematical equations to show the calculations).
   (a) Centering or mean removal
   (b) Scaling of a feature to a range $[a, b]$
   (c) Standardization
   (d) Normalization (between 0 and 1)
2. Apply the above operations to the following data matrix with 5 samples and two features independently and show the processed data matrix. You will have 4 different matrices based on each processing operation. For scaling pick $a = 0$ and $b = 1$.

$$\boldsymbol{X} = \begin{bmatrix} 1 & 2 \\ -1 & 1 \\ 0 & 1 \\ 2 & 4 \\ 3 & 1 \end{bmatrix}$$

[1] For an implementation of these operation in scikit-learn please see this.