Recitation 6

RL & Unsupervised Learning

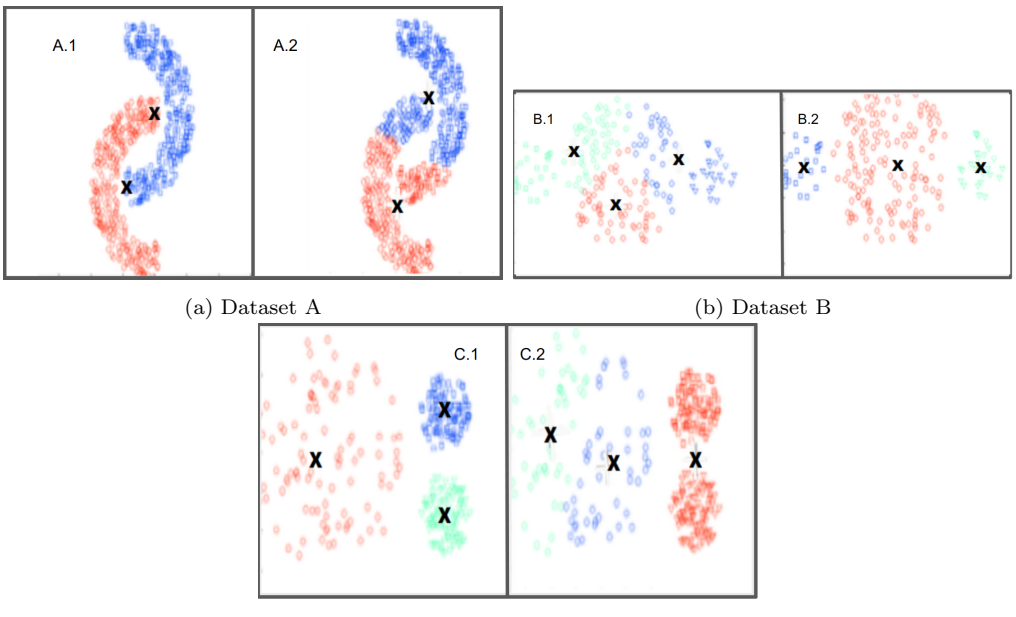
10-701: Introduction to Machine Learning

March 27, 2026

1 K-Means

1.1 Visualization

Consider the 3 datasets A, B and C as shown in Figure 1. Each dataset is classified into K clusters as represented by different colors in the figure. For each dataset, determine which image with cluster centers (denoted by **X**) is generated by K-means method. The distance measure used here is the Euclidean distance.



(a) Dataset A

(b) Dataset B

(c) Dataset C

Figure 1: Datasets

1.2 Practical Example

Consider a dataset $\mathcal{D}$ with 5 points as shown below. Perform a K-means clustering on this dataset with $K = 2$ using the Euclidean distance as the distance function.

Remember that in the K-means algorithm, an iteration consists of performing following tasks: assigning each data point to it's nearest cluster center followed by recomputing those centers based on all the data points assigned to it.

Initially, the 2 cluster centers are chosen as $\boldsymbol{\mu}_1 = [5.3, 3.5]^T$, $\boldsymbol{\mu}_2 = [5.1, 4.2]^T$.

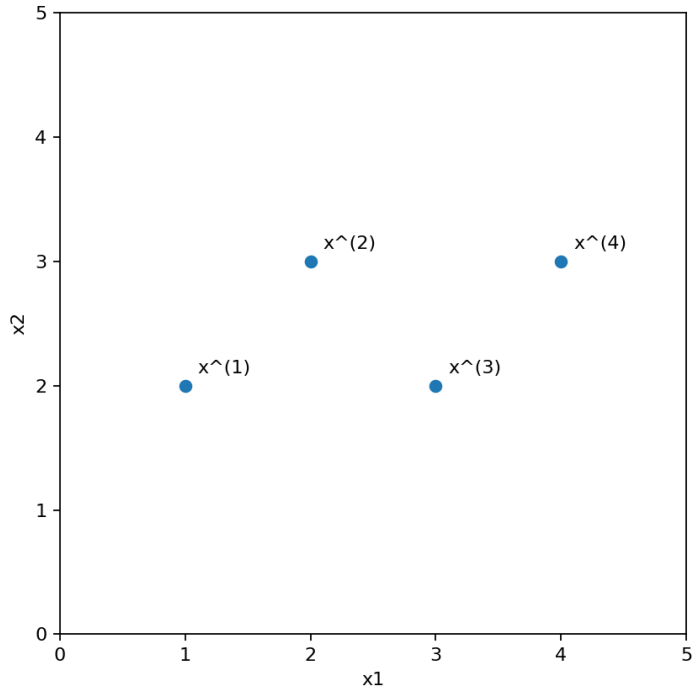
$$\mathcal{D} = \begin{bmatrix} 5.5 & 3.1 \\ 5.1 & 4.8 \\ 6.3 & 3.0 \\ 5.5 & 4.4 \\ 6.8 & 3.5 \end{bmatrix}$$

1. What will be the coordinates of the center for cluster 1 after the first iteration?
2. What will be the coordinates of the center for cluster 2 after the first iteration?
3. How many points will belong to cluster 1 after the first iteration?
4. How many points will belong to cluster 2 after the first iteration?
5. In general (not directly pertaining to this example), is it possible to have empty clusters, once the algorithm converges? Is K-means guaranteed to converge?

2 PCA

2.1 Data

Consider dataset $\mathcal{D} = \{\mathbf{x}^{(1)} = \begin{bmatrix}1\\2\end{bmatrix}, \mathbf{x}^{(2)} = \begin{bmatrix}2\\3\end{bmatrix}, \mathbf{x}^{(3)} = \begin{bmatrix}3\\2\end{bmatrix}, \mathbf{x}^{(4)} = \begin{bmatrix}4\\3\end{bmatrix}\}$. A visualization of the dataset is as below.



2.2 Centering Data

Centering is crucial for PCA. We must preprocess data so that all features have zero mean before applying PCA, i.e.

$$\frac{1}{N}\sum_{i=1}^{N}\mathbf{x}^{(i)} = \vec{0}$$

Compute the centered dataset:

$\mathbf{x}^{(1)} =$ _____ $\mathbf{x}^{(2)} =$ _____

$\mathbf{x}^{(3)} =$ _____ $\mathbf{x}^{(4)} =$ _____

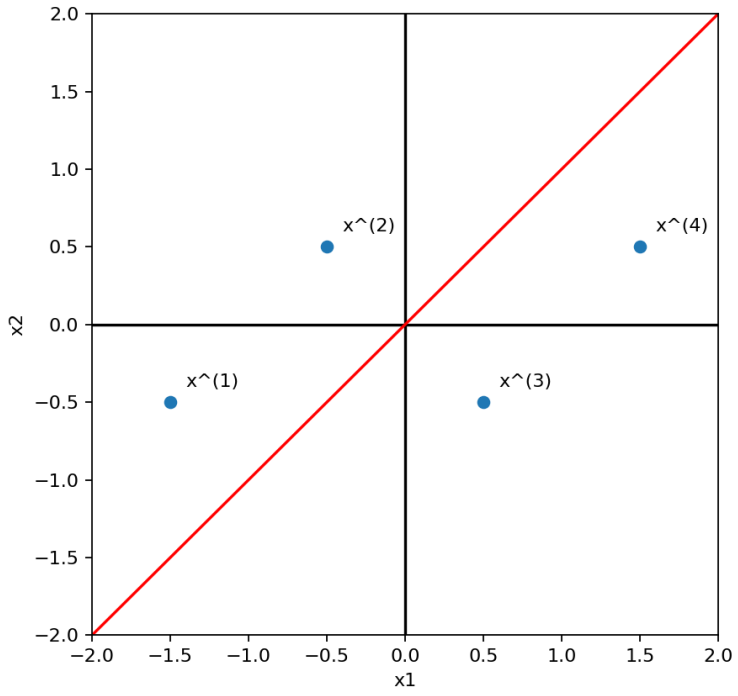
2.3 Unit vector

In order to easily compute the projected coordinates of data, we need to make the projected directions unit vectors. Suppose we want to project our data onto the vector $\mathbf{v} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$. Normalize $\mathbf{v}$ to be a unit vector.

$\mathbf{v} =$ _____

2.4 Project Data

The centered data should now look like the following:



Suppose we want to project the centered data onto $\mathbf{v}$, where $\mathbf{v}$ goes through the origin. Compute the magnitude of the projections, i.e. compute $z^{(i)} = \mathbf{v}^T\mathbf{x}^{(i)}, \forall 1 \leq i \leq N$.

$z^{(1)} =$ _____ $z^{(2)} =$ _____

$z^{(3)} =$ _____ $z^{(4)} =$ _____

Let $\mathbf{x}^{(i)'}$ be the projected point of $\mathbf{x}^{(i)}$. Note that $\mathbf{x}^{(i)'} = \mathbf{v}^T\mathbf{x}^{(i)}\mathbf{v} = z^{(i)}\mathbf{v}$. Compute the projected coordinates:

$\mathbf{x}^{(1)'} =$ _____ $\mathbf{x}^{(2)'} =$ _____

$\mathbf{x}^{(3)'} =$ _____ $\mathbf{x}^{(4)'} =$ _____

2.5 Reconstruction Error

One of the two goals of PCA is to find new directions to project our dataset onto such that it **minimizes the reconstruction error**, where the reconstruction error is defined as following:

$$\text{Reconstruction Error} = \frac{1}{N}\sum_{i=1}^{N} \|\mathbf{x}^{(i)'} - \mathbf{x}^{(i)}\|_2^2$$

What is the reconstruction error in our case?

Reconstruction Error = _____

2.6 Variance of Projected Data

Another goal is to find new directions to project our dataset onto such that it **maximizes the variance of the projections**, where the variance of projections is defined as following:

$$\begin{aligned}
\text{variance of projection} &= \frac{1}{N}\sum_{i=1}^{N}(z^{(i)} - \hat{E}[z])^2 \\
&= \frac{1}{N}\sum_{i=1}^{N}(\mathbf{v}^T\mathbf{x}^{(i)} - \frac{1}{N}\sum_{j=1}^{N}\{\mathbf{v}^T\mathbf{x}^{(j)})^2 \\
&= \frac{1}{N}\sum_{i=1}^{N}(\mathbf{v}^T\mathbf{x}^{(i)} - \mathbf{v}^T(\frac{1}{N}\sum_{j=1}^{N}\mathbf{x}^{(j)}))^2 \\
&= \frac{1}{N}\sum_{i=1}^{N}(\mathbf{v}^T\mathbf{x}^{(i)} - \mathbf{v}^T\vec{0})^2 \\
&= \frac{1}{N}\sum_{i=1}^{N}(\mathbf{v}^T\mathbf{x}^{(i)})^2
\end{aligned}$$

What is the variance of the projections?

variance = _____

2.7 Principal component of PCA

What is the first principal component of $X = [x^{(1)}, x^{(2)}, x^{(3)}, x^{(4)}]^T$?

3 Introduction to Reinforcement Learning

Recall that a Markov decision process is a tuple $(\mathcal{S}, \mathcal{A}, T, R, \gamma, s_0)$, where:

- $\mathcal{S}$ is the set of states
- $\mathcal{A}$ is the set of actions
- $T : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$ is the transition function
- $R : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R}$ is the reward function
- $\gamma \in [0, 1)$ is the discount factor
- s_0 is the start state

A **policy** is any function $\pi : \mathcal{S} \to \mathcal{A}$ mapping from the states to the actions. We say that we are executing some policy π if, whenever we are in state s, we take action $a = \pi(s)$.

For the optimal policy function π^* we can compute its **value function** at state s as:

$$\begin{aligned} V^{\pi^*}(s) &= V^*(s) \\ &= \mathbb{E}\left[R(s_0, \pi^*(s_0), s_1) + \gamma R(s_1, \pi^*(s_1), s_2) + \gamma^2 R(s_2, \pi^*(s_2), s_3) \cdots \mid s_0 = s, \pi^*\right]. \end{aligned}$$

In other words, this is the best possible expected total payoff that can be attained using **any policy** π.

This **optimal value function** can be represented using Bellman's equations (named **Bellman optimality equation** for state value function), i.e.,

$$V^*(s) = \max_{a \in \mathcal{A}} \sum_{s' \in \mathcal{S}} p(s'|s, a)(R(s, a, s') + \gamma V^*(s')).$$

If $R(s, a, s') = R(s, a)$ (deterministic reward), then we have the form we saw in class:

$$V^*(s) = \max_{a \in \mathcal{A}} \left\{ R(s, a) + \gamma \sum_{s' \in \mathcal{S}} p(s'|s, a) V^*(s') \right\}.$$

The interpretation of Bellman equation is also very intuitive: "how valuable is the current state under a policy?" Then it naturally follows that the optimal policy $\pi^* : \mathcal{S} \to \mathcal{A}$ can be found as

$$\pi^*(s) = \arg\max_{a \in \mathcal{A}} \sum_{s' \in \mathcal{S}} p(s'|s, a)(R(s, a, s') + \gamma V^*(s')).$$

4 Value Iteration

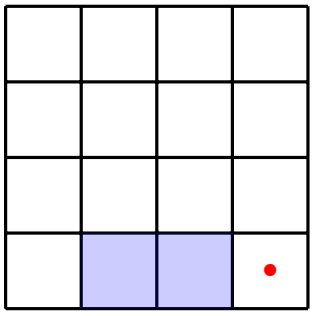


Figure 2: The cliff-walking environment

Here, we assume a 4×4 grid environment. The reward is 1 for reaching the cell with the red dot, -1 for reaching the shaded cells, and 0 for all other cells. The episode ends if the agent lands in either the shaded cells or the red-dot cell. The state space ($\mathcal{S}$) is all the cells. The action space (A) is up, down, left or right. If the agent tries to move out of the grid, it simply goes back to its previous state. The discount factor, γ, is 0.9.

Bellman optimality equation for state value function:

$$V^*(s) = \max_{a \in \mathcal{A}} \sum_{s' \in \mathcal{S}} p(s'|s,a)(R(s,a,s') + \gamma V^*(s'))$$

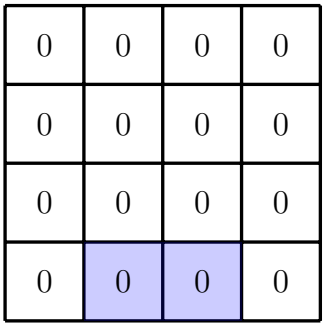
We can numerically approximate V^* using synchronous value iteration and asynchronous value iteration. The recurrence relation is defined as follows for **synchronous** value iteration for all $s \in \mathcal{S}$:

$$V^{(t+1)}(s) = \max_{a \in \mathcal{A}} \sum_{s' \in \mathcal{S}} p(s'|s,a)(R(s,a,s') + \gamma V^{(t)}(s')).$$

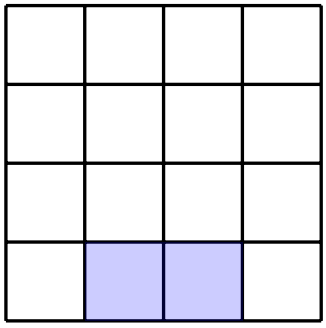
Notice the time term (t), hence the name synchronous. For **asynchronous** value iteration, we use for all $s \in \mathcal{S}$:

$$V(s) = \max_{a \in \mathcal{A}} \sum_{s' \in \mathcal{S}} p(s'|s,a)(R(s,a,s') + \gamma V(s')).$$

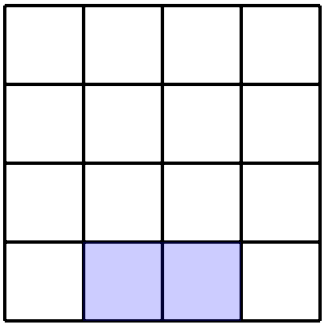
1. Find the updated value of each cell after the first round and after the second round of synchronous value iteration. Assume deterministic transition function.



Initial

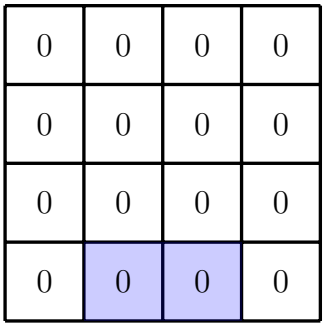


After 1st round

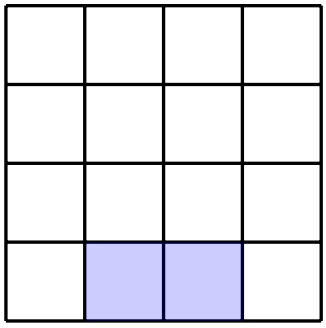


After 2nd round

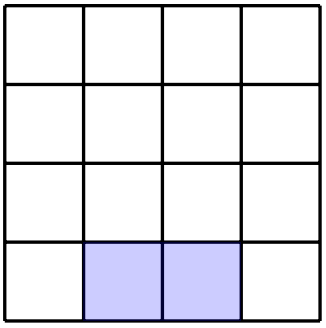
2. Starting over, find the update value of each cell after one round of asynchronous value iteration. Visit each cell in two different orders: (1) bottom right to top left, and (2) top left to bottom right.



Initial

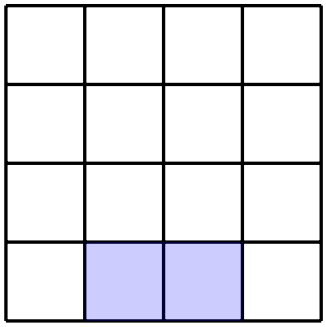


BR to TL

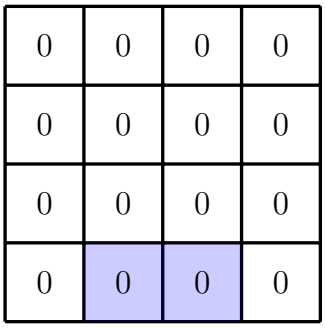


TL to BR

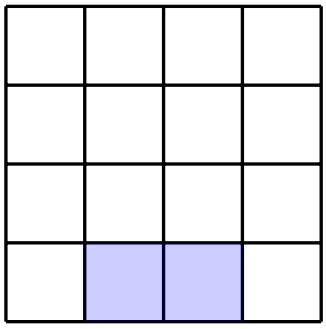
3. What is the policy learned after the one round of asynchronous value iteration when the cells were visited from bottom right to top left? If there is a tie, pick the action that comes first alphabetically (i.e., priority: $\downarrow > \leftarrow > \rightarrow > \uparrow$).



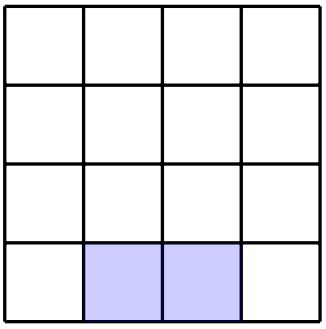
4. Now suppose that the environment is sloped downward towards the cliff, with all the other settings unchanged. For every action taken, there is a 0.5 probability that the agent will move as intended and a 0.5 probability that the agent will slip and move 1 cell down instead. Report the values after two rounds of synchronous value iteration.



Initial



After 1st round



After 2nd round

5 Q-Learning

5.1 Q-Value

We can also define a Q function $Q(s, a)$ that returns the expected discounted future value of taking action a at state s.

Question: When would we want to use $Q(s, a)$ instead of $V(s)$?
Answer:

We can take a part of $V^*(s)$ as $Q^*(s, a)$ so that we have

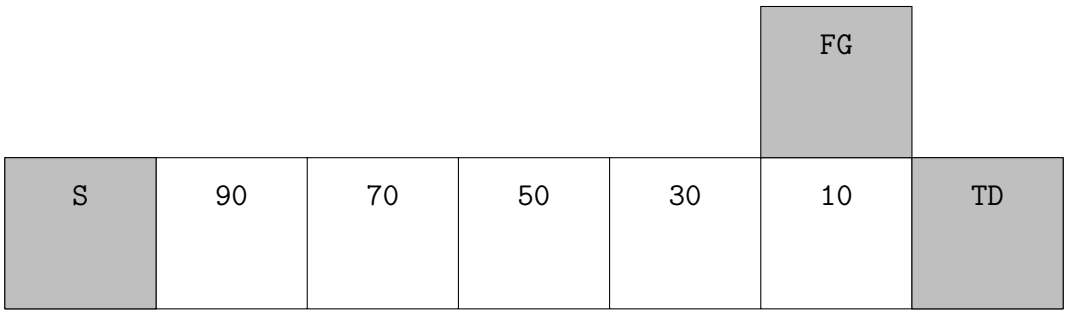
$$Q^*(s,a) = \sum_{s' \in \mathcal{S}} p(s'|s,a)(R(s,a,s') + \gamma V^*(s'))$$

$$V^*(s) = \max_{a \in \mathcal{A}} \underbrace{\sum_{s' \in \mathcal{S}} p(s'|s,a)(R(s,a,s') + \gamma V^*(s'))}_{Q^*(s,a)} = \max_{a \in \mathcal{A}} Q^*(s,a).$$

Then it follows that the optimal policy can be obtained as

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

Let's play some football (the ~~wrong~~ American kind). We will divide the field into eight states (labeled slightly differently from an official football field for clarity):



Here are some basic rules (slightly different than official football rules, but let's work with it for the sake of this example):

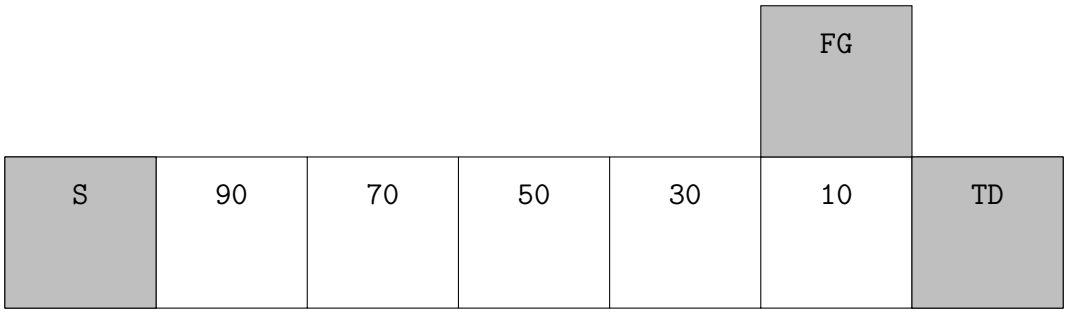
- There are two teams. For simplicity, let's call them Offense and Defense.
- The length of the field is 100 yards; we will set each 20-yard interval on this field as its own state, in addition to the end zones (`S`, `TD`, and `FG`).
- The drive (or, in reinforcement learning terms, episode) terminates if the ball is moved into the end zone.
- At each state, the Offense may either `run` the ball or `pass` it (always in the direction of `TD`).

The head coach argues that it is not a good idea to assume a value for the transition probability from one state to the next. She also argues that since it is not the case that the Offense will always score seven points when going into the `TD` state (they can also score six points or eight) it is better to not assume a reward function. It is also possible to start the drive from anywhere in the field. For all these reasons, she encourages the offensive coordinators to try Q-learning.

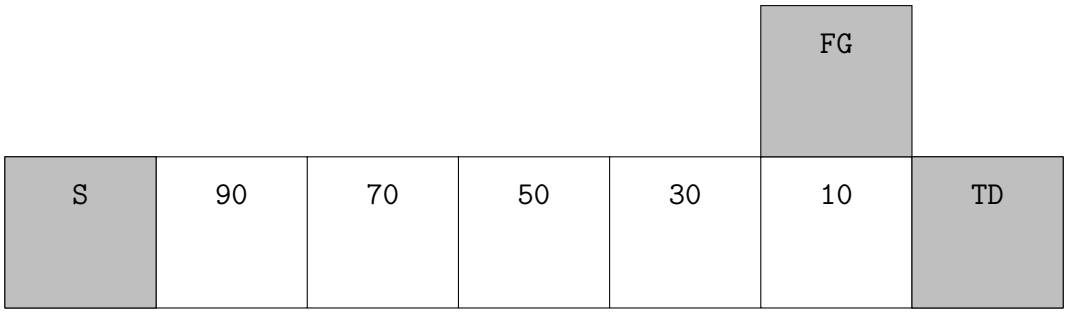
1. The offensive coordinators, now convinced, decide to build a Q-value table that matches the state and action space. What is the size of this table?

2. The offensive coordinators initialize $Q(s, a) = 0 \ \forall s, a$. They decide to update it by sampling random drives from games previously played by the Offense this season (note that this approach uses experience replay and not a standard MDP since individual plays in a football game are not independent unlike moves in a grid world). They give you the empty Q-value table for ease:

   Q($\cdot$, `run`):

   Q($\cdot$, `pass`):

   Time to watch some football :) Assume the learning rate $\alpha = 0.1$ and the discount factor $\gamma = 0.5$. Update the Q-table above for each of the following episodes using the Temporal Difference error update:

   (a) **Episode 1, iteration 1**: Beginning at the `30` state, Offense `run`s the ball to `10`. Offense collects zero reward.

$Q(\texttt{30}, \texttt{run}) =$

(b) **Episode 1, iteration 2**: Offense passes the ball from the 10 but stays at the 10. Offense collects zero reward.
$Q(\texttt{10}, \texttt{pass}) =$

(c) **Episode 1, iteration 3**: From the 10, Offense passes the ball again and reaches the TD state. The drive terminates. Offense collects a reward of 7.
$Q(\texttt{10}, \texttt{pass}) =$

(d) After updating the Q-function for the first episode, the head coach decides to quiz the offensive coordinators. If the Offense is in state 10, what is the best action?
$\pi(\texttt{10}) =$

(e) **Episode 2, iteration 1**: Beginning at the 90 state, the Offense runs the ball, causing a safety (they enter the S state). The drive terminates. Offense collects -2 reward.
$Q(\texttt{90}, \texttt{run}) =$

(f) After updating the Q-function for the second episode, the head coach decides to quiz the offensive coordinators again. If the Offense is in state 90, what is the best action to take?
$\pi(\texttt{90}) =$

(g) **Episode 3, iteration 1**: Beginning at 50, the Offense passes the ball but stays at the 50. Offense collects zero reward.
$Q(\texttt{50}, \texttt{pass}) =$

(h) **Episode 3, iteration 2**: Beginning at the 50, the Offense passes the ball to the 30. Offense collects zero reward.
$Q(\texttt{50}, \texttt{pass}) =$

(i) **Episode 3, iteration 3**: Beginning at the 30, the Offense runs the ball to the 10. Offense collects zero reward.
$Q(\texttt{30}, \texttt{run}) =$

(j) **Episode 3, iteration 4**: Beginning at the 10, the Offense runs the ball and ends the drive after entering the FG state. Offense collects a reward of 3 points.
$Q(\texttt{10}, \texttt{run}) =$

(k) After updating the Q-function for the third episode, the head coach decides to quiz the offensive coordinators for the last time. If the Offense is in state 10, what is the best action to take?
$\pi(\texttt{10}) =$