Reminders

- PRA3 due 2/22
- Course project progress report 1 due 2/27
- HW3 due 2/29
- Come to OH for course project discussion!

Artificial Intelligence Methods for Social Good

Lecture 11

Case Study: Assist Non-Profits in Improving Maternal and Child Health

17-537 (9-unit) and 17-737 (12-unit)

Instructor: Fei Fang

feifang@cmu.edu

Outline

- Assist Non-Profits in Improving Maternal and Child Health

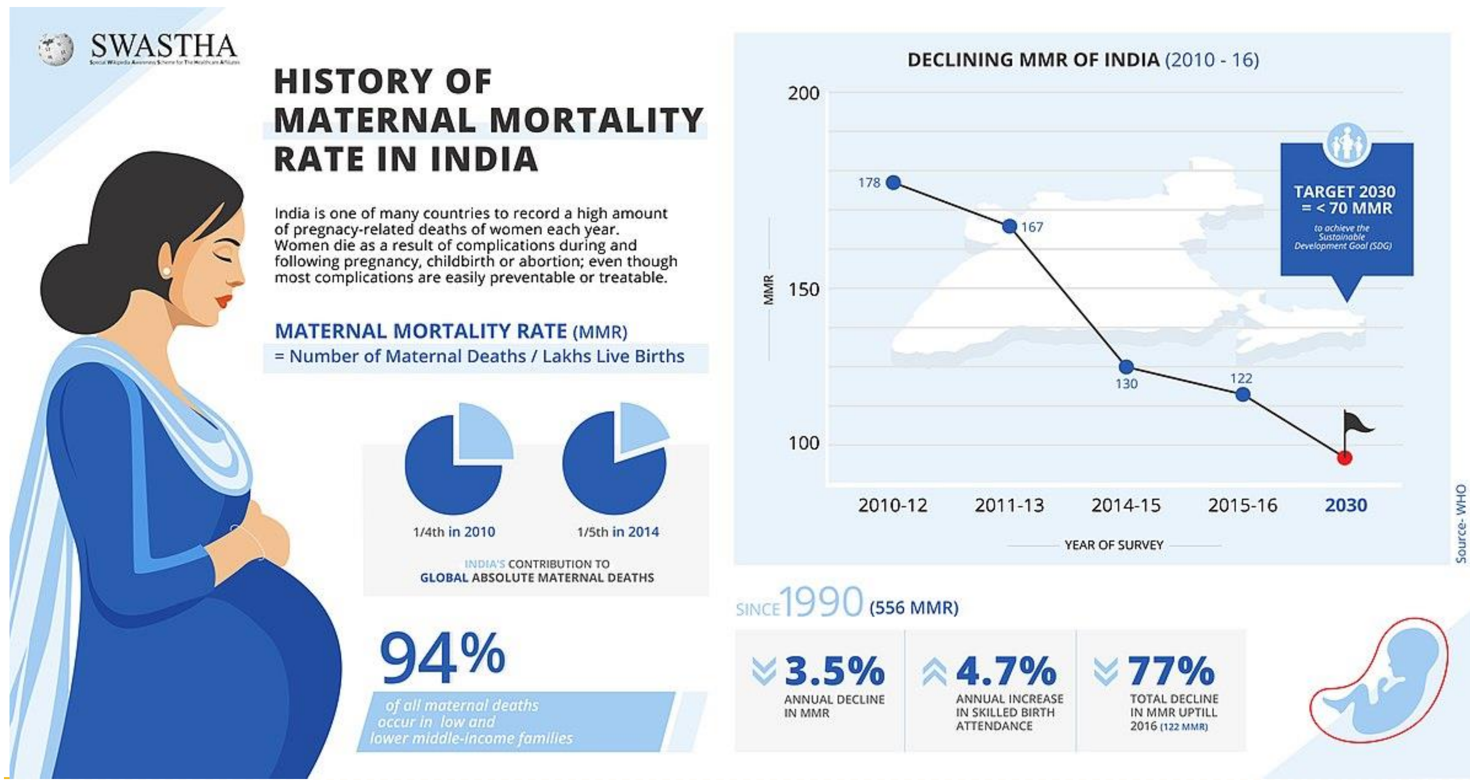
- Monte-Carlo Tree Search and Applications to Healthcare (Optional for this lecture)

Learning Objectives

- Understand the concept of
  - K-means clustering
  - Monte-Carlo Tree Search
- For the application problem, briefly describe
  - Significance/Motivation
  - Task being tackled
  - AI method used
  - Evaluation process and criteria

Deliver Critical Health Info to New and Expecting Moms

- Almost 90 percent of maternal deaths in India are avoidable if women receive the right kind of intervention



Deliver Critical Health Info to New and Expecting Moms

mMitra

mMitra is a free mobile voice call service by ARMMAN that sends timed and targeted preventive care information weekly/bi-weekly directly to the phones of the enrolled women through pregnancy and infancy in their chosen language and timeslot.

2.80 Million	9	40	97
WOMEN REACHED	STATES	NGO PARTNERS	HOSPITALS

Deliver Critical Health Info to New and Expecting Moms

- Automated voice message/text sent
  - Average voice message length: 1 min
- But do beneficiaries listen? Does the nonprofit know?
  - They know how long the beneficiaries stay on
- How to keep beneficiaries engaged?

Deliver Critical Health Info to New and Expecting Moms

- Non-profits can initiate service calls!
- But…limited health-worker time available
- Each week, health workers can call a limited number of beneficiaries

Deliver Critical Health Info to New and Expecting Moms

- Task: Choose a subset of M beneficiaries to call each week

- Simple solution: Round Robin

- Can we do better with AI?

Mate et al, 2021. Field Study in Deploying Restless Multi-Armed Bandits: Assisting Non-Profits in Improving Maternal and Child Health

Recap: Restless Multi-Armed Bandit

- N arms
- Each arm is a 2-action MDP
  - In time t, an arm is in some state
  - Two possible actions: active (pull the arm), or passive (not pull the arm)
  - Action brings the planner reward and brings the arm into a successor state in time $t + 1$
  - “Restless”: state transition happens even if the arm is not pulled
- Planner can observe the state of each arm, and pull αN arms in each time step

Paired Discussion

- Formulate the following problem as an RMAB problem: choosing a subset of K beneficiaries to call each week

- What is an arm in this context?
- What is one time step in this context?
- What does active and passive action mean?
- What are the states for the MDP of each arm?
- What is the reward for each state/action?
- How to estimate the state transition probabilities?

RMAB Formulation

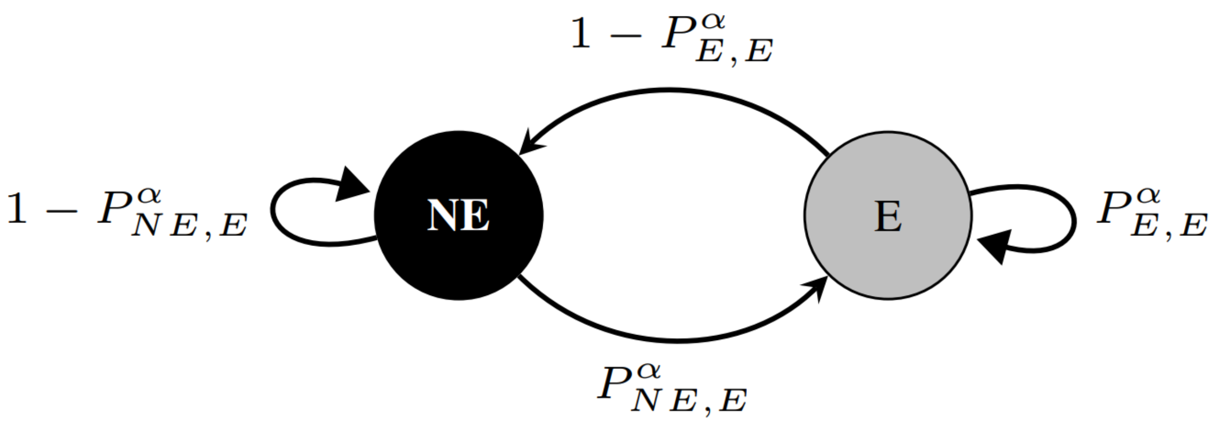
- An arm: a beneficiary
- A time step: a week
- Active action (a): make a service call to the beneficiary
- Passive action (p): no call to the beneficiary

RMAB Formulation

- States: Engaging (E) and Not Engaging (NE)
  - State is known to the planner
  - If a beneficiary stays on the automated voice message for more than 30 seconds in a week, she is in state E
  - Otherwise, she is in state NE
- Reward: 1 if in state E, 0 if in state NE

RMAB Formulation

- Transition probabilities (different for different arms)
  - α can be a (active) or p (passive)



- Question: Intuitively, what is the (inequality) relationship between $P^{a}_{NE,E}$ and $P^{p}_{NE,E}$? How about $P^{a}_{E,E}$ and $P^{p}_{E,E}$?

Key Challenge: Estimate Transition Probabilities

- Historical Data D_{train}
- Each data point (f, E) corresponds one beneficiary
  - f is a static feature vector for the beneficiary: age, education level, income bracket, phone owner in the family, gestation age, number of children, preferred language, preferred slots for receiving voice messages
  - E is an episode storing the trajectory of (s, α, s') pairs for that beneficiary

- In the case with ARMMAN: 4238 beneficiaries who enrolled into the program between May-July 2020

Key Challenge: Estimate Transition Probabilities

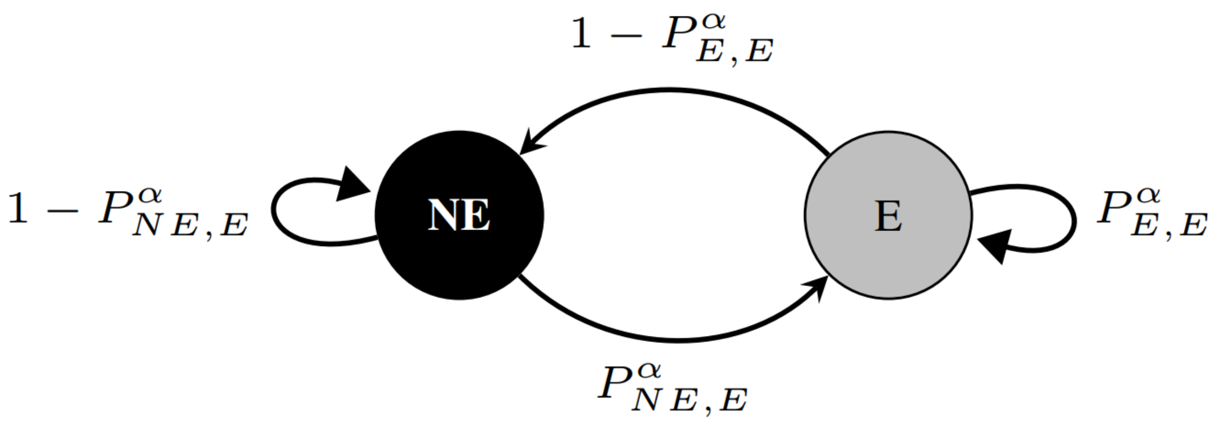
- If we have sufficient historical data for each arm (each beneficiary): compute the empirical transition probabilities $P^a_{NE,E}$, $P^p_{NE,E}$, $P^a_{E,E}$ and $P^p_{E,E}$

- But…a beneficiary stays the program only during / around pregnancy and for a short period of time after delivery

- Impossible to get sufficient data to estimate the probabilities for each arm!

Key Challenge: Estimate Transition Probabilities

- Discuss: Why the following simple approach does not work in this problem?
  - Assume all arms share the same transition probabilities
  - Use historical data of all the beneficiaries to estimate the probabilities



Key Idea: Clustering

- Divide the beneficiaries into clusters
- Each cluster should consist of beneficiaries who are similar to each other, and we assume that they have the same transition probabilities
- For each cluster, use data from all of the beneficiaries in the cluster to estimate the probabilities

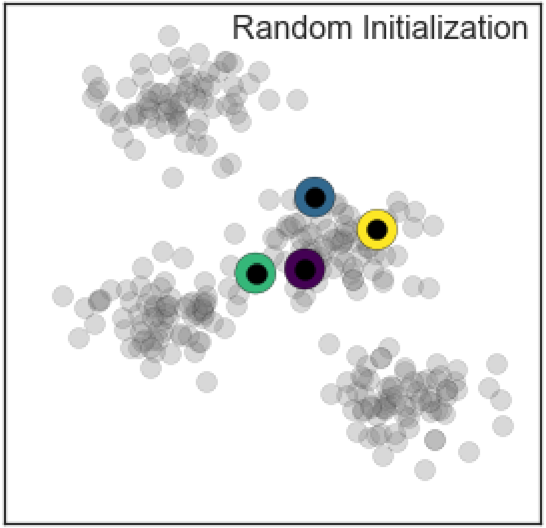
Clustering

- Clustering is a typical machine learning task
  - Unsupervised learning: learn from unlabeled data
  - Divide unlabeled data points into clusters
  - Given a dataset with each data point described by a feature vector, divide the data points into k clusters such that data points in the same cluster are similar to each other

- Commonly used approaches
  - K-Means Clustering
  - Gaussian Mixture Models

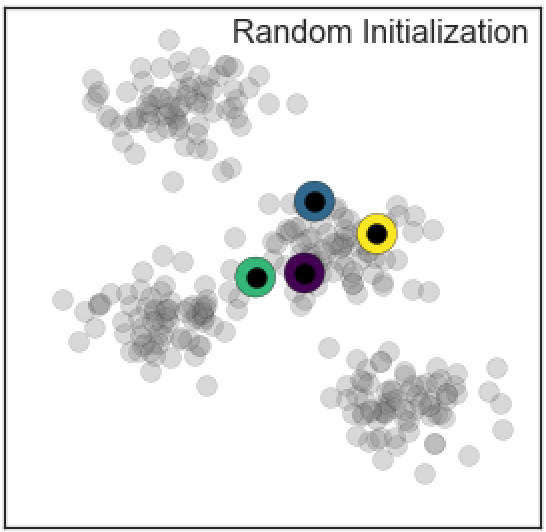
K-Means Clustering

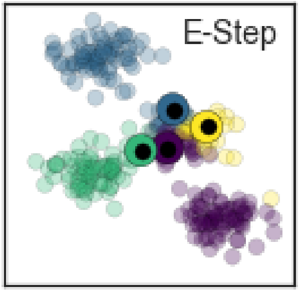
- Start with K random points, treat them as the initial cluster center



K-Means Clustering

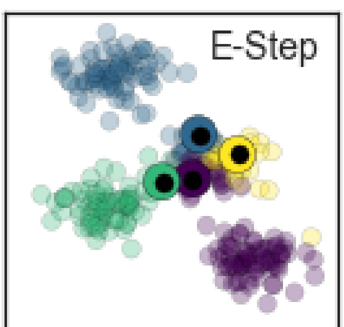
- Start with K random points, treat them as the initial cluster center
- Expectation step (E-step): For each data point, assign it to the nearest cluster center

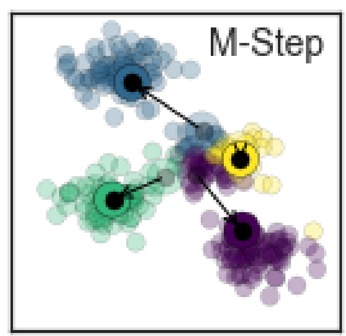




K-Means Clustering

- Start with K random points, treat them as the initial cluster center
- Expectation step (E-step): For each data point, assign it to the nearest cluster center
- Maximization step (M-step): For each cluster, set the new cluster center to the mean/centroid of all the data points in the cluster





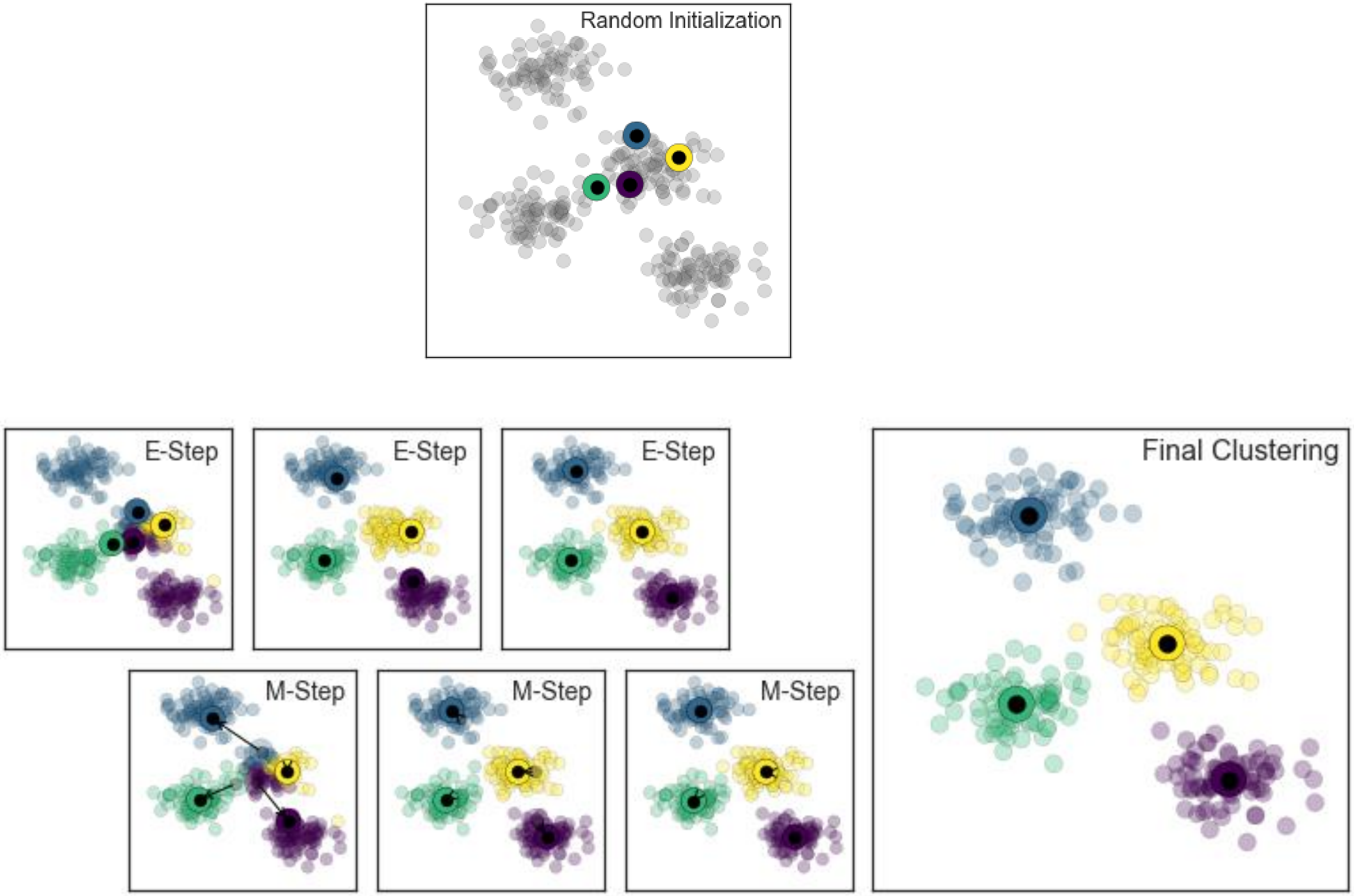
https://colab.research.google.com/github/jakevdp/PythonDataScienceHandbook/blob/master/notebooks/05.11-K-Means.ipynb

K-Means Clustering

- Start with K random points, treat them as the initial cluster center
- Expectation step (E-step): For each data point, assign it to the nearest cluster center
- Maximization step (M-step): For each cluster, set the new cluster center to the mean/centroid of all the data points in the cluster
- Repeat E-step and M-step until the assignment of the data points do not change anymore

https://colab.research.google.com/github/jakevdp/PythonDataScienceHandbook/blob/master/notebooks/05.11-K-Means.ipynb

K-Means Clustering



https://colab.research.google.com/github/jakevdp/PythonDataScienceHandbook/blob/master/notebooks/05.11-K-Means.ipynb

K-Means Clustering

- How to measure the quality of clustering?
  - Use some error function
  - Option 1: sum of squared errors

$$\sum_{k=1}^{K} \sum_{i:i \text{ in cluster } k} \left(f_i - c^k\right)^2$$

  - Option 2: Root Mean Square Error (RMSE)

$$\sqrt{\frac{\sum_i \left(f_i - \widehat{f_i}\right)^2}{N}}$$

  - Higher K $\rightarrow$ Lower error

K-Means Clustering

- K-means clustering in practice

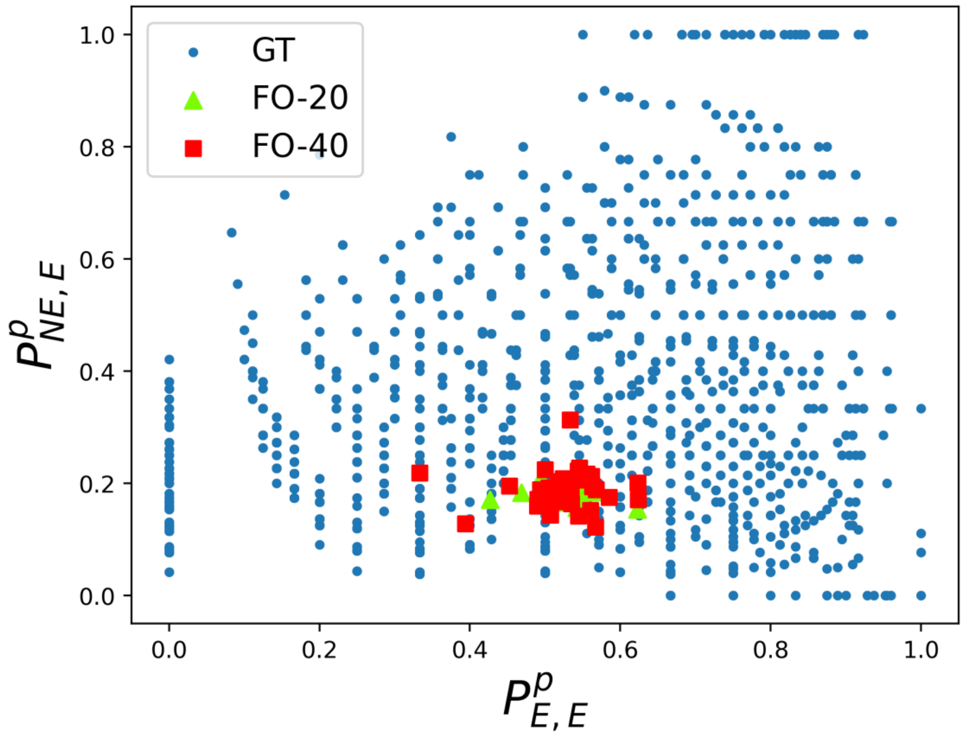
```
[ ]  from sklearn.cluster import KMeans
     kmeans = KMeans(n_clusters=4)
     kmeans.fit(X)
     y_kmeans = kmeans.predict(X)
```

Divide unlabeled dataset X into 4 clusters

https://colab.research.google.com/github/jakevdp/PythonDataScienceHandbook/blob/master/notebooks/05.11-K-Means.ipynb

K-Means Clustering for Our Problem

- Approach 1: Features-Only clustering (FO)
  - Directly use static features f for clustering
  - Reasonable but not performing well



K-Means Clustering for Our Problem

- Approach 2: Feature + All Probabilities (FAP)
  - Two-level approach
  - Step 1: use a rule-based method to divide beneficiaries into (a large number of) buckets
  - Step 2: estimate transition probabilities for each bucket
  - Step 3: use the estimated transition probabilities as features for each bucket, cluster the buckets into K clusters

- Approach 3: Feature + Passive Probabilities
  - Same as Approach 2 (FAP), except that it only uses passive action probabilities in step 3 clustering

K-Means Clustering for Our Problem

- Approach 4: Passive Transition-Probability based Clustering (PPF)
  - Step 1: For each beneficiary, use historical data to estimate transition probabilities for passive action
  - Step 2: Use k-means clustering to identify cluster centers
  - Step 3: learn a map from feature vector f to cluster assignment (now a supervised learning problem!) using random forest

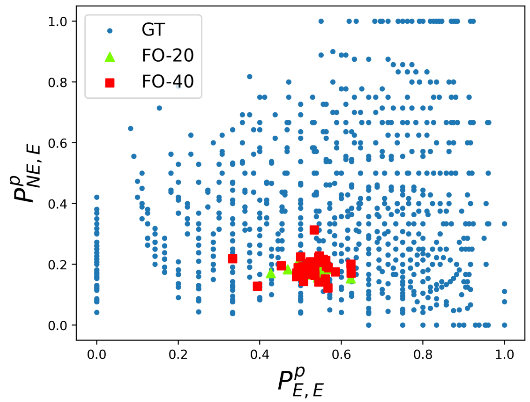
Evaluation of Clustering Methods

- Compare RMSE and standard deviation of cluster sizes
  - We want low error and balanced clusters sizes
  - Tested 20 and 40 clusters

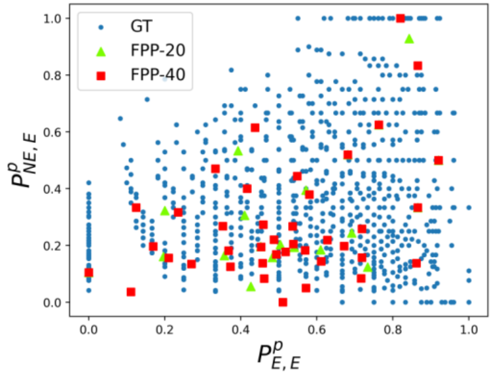
Clustering Method	Average RMSE		Standard Deviation	
	k = 20	**k = 40**	**k = 20**	**k = 40**
FO	0.229	0.228	143.30	74.22
FPP	0.223	0.222	596.19	295.01
FAP	0.224	0.223	318.46	218.37
PPF	0.041	0.027	145.59	77.50

Evaluation of Clustering Methods

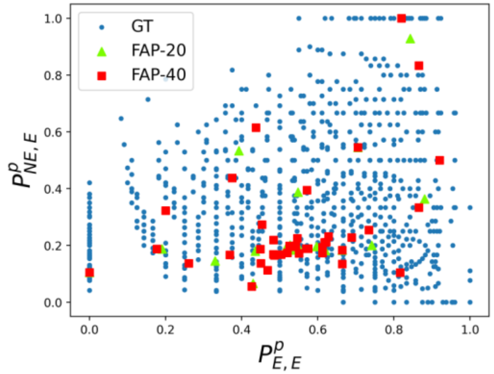
- Visualization of cluster centers



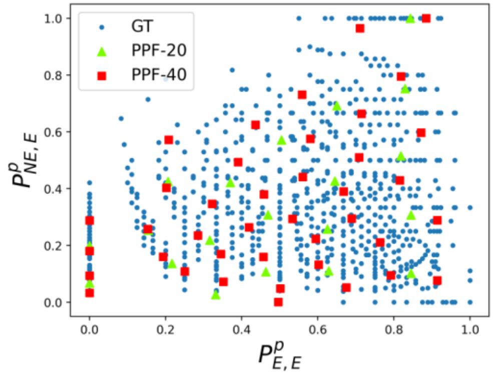
Features-Only Clustering



(b) FPP clustering



(c) FAP clustering



(d) PPF clustering

Select Beneficiaries in Each Time Step

- Recap: Whittle Index: Infimum subsidy that makes the planner indifferent between the "active" and the "passive" actions

$$W(s) = \inf_{\lambda}\{\lambda: Q_\lambda(s, passive) = Q_\lambda(s, active)\}$$

- Choose the arms with highest $W(s)$

Experimental Study

- Compare the following cases
  - Control group (Current Standard of Care): Beneficiaries could initiate a service call with trained health workers by request, but no calls initiated by the non-profits
  - RMAB-based method: use 40 clusters, PPF clustering
  - Alternative method: Round Robin
- How to run a field study?

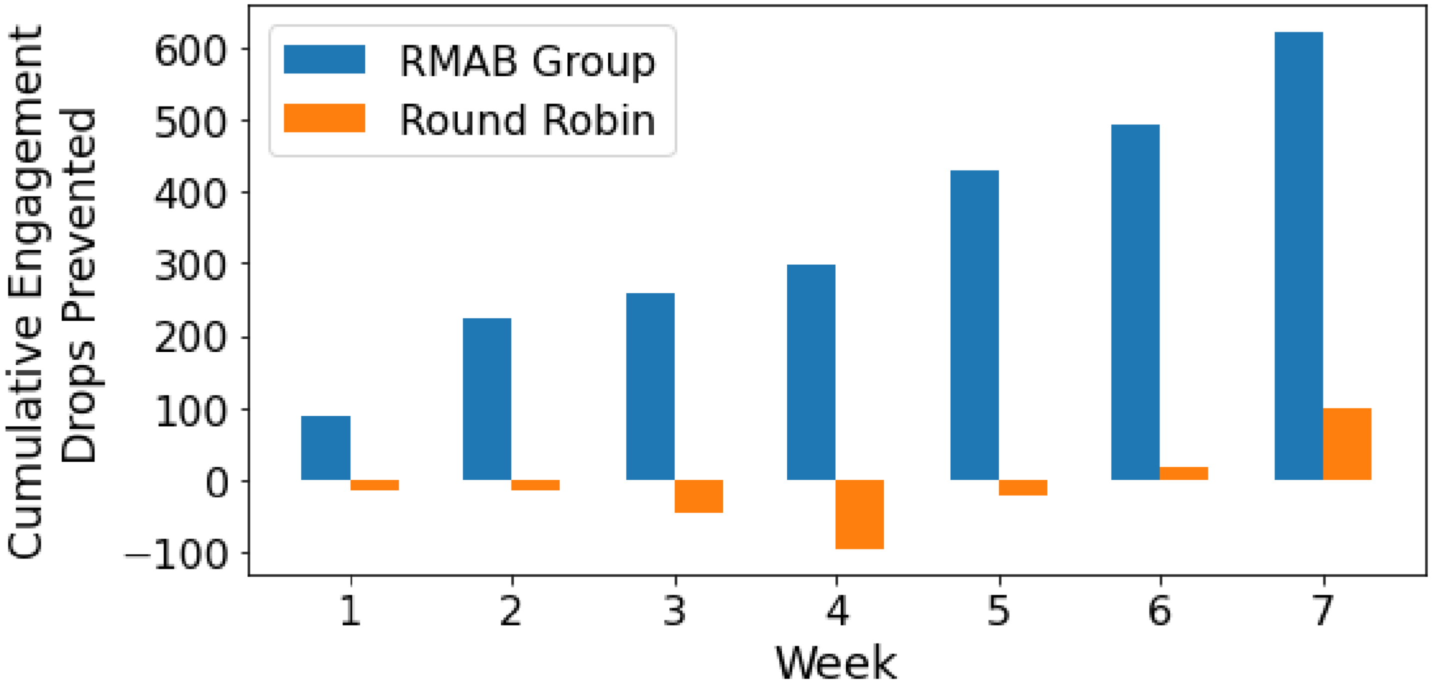
Experimental Study

- D_{test}: beneficiaries registered in the program between Feb 16, 2021 and March 15, 2021
- 23003 beneficiaries in total
  - Randomly distributed across 3 groups: CSOC group, Round Robin group, RMAB group
- State distribution in week 0 of the study (Apr 19 - Apr 26, 2021)

Group	Engaging (E)	Non-Engaging (NE)	Total
RMAB	3571	4097	7668
RR	3647	4021	7668
CSOC	3661	4006	7667

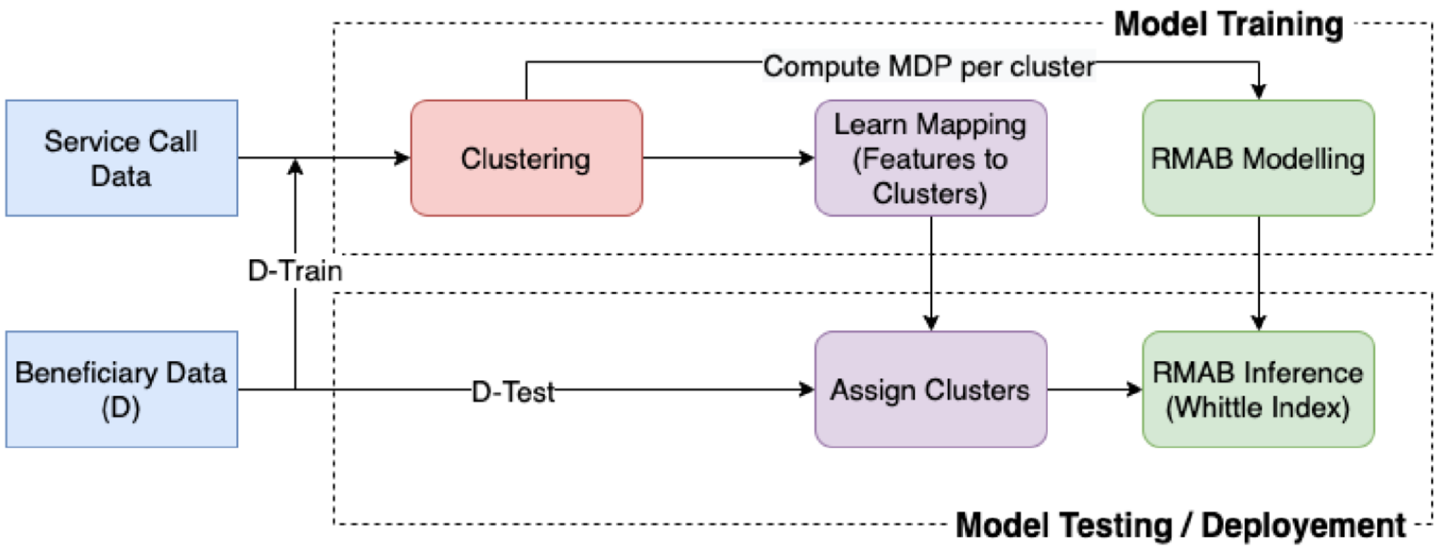
Experimental Study

- RMAB method leads to more engagement drops prevented!



Summary

- Overall Pipeline



Learning Objectives

- Understand the concept of
  - K-means clustering
  - Monte-Carlo Tree Search
- For the application problem, briefly describe
  - Significance/Motivation
  - Task being tackled
  - AI method used
  - Evaluation process and criteria

Monte Carlo Tree Search

- General framework to make online decision in sequential decision making problems
  - E.g., online planning in MDPs, to determine game plays in Go, chess, video games etc
- Not only applicable to MDPs, but also other domains that cannot be modeled as MDPs
  - The idea of Q value can still be used

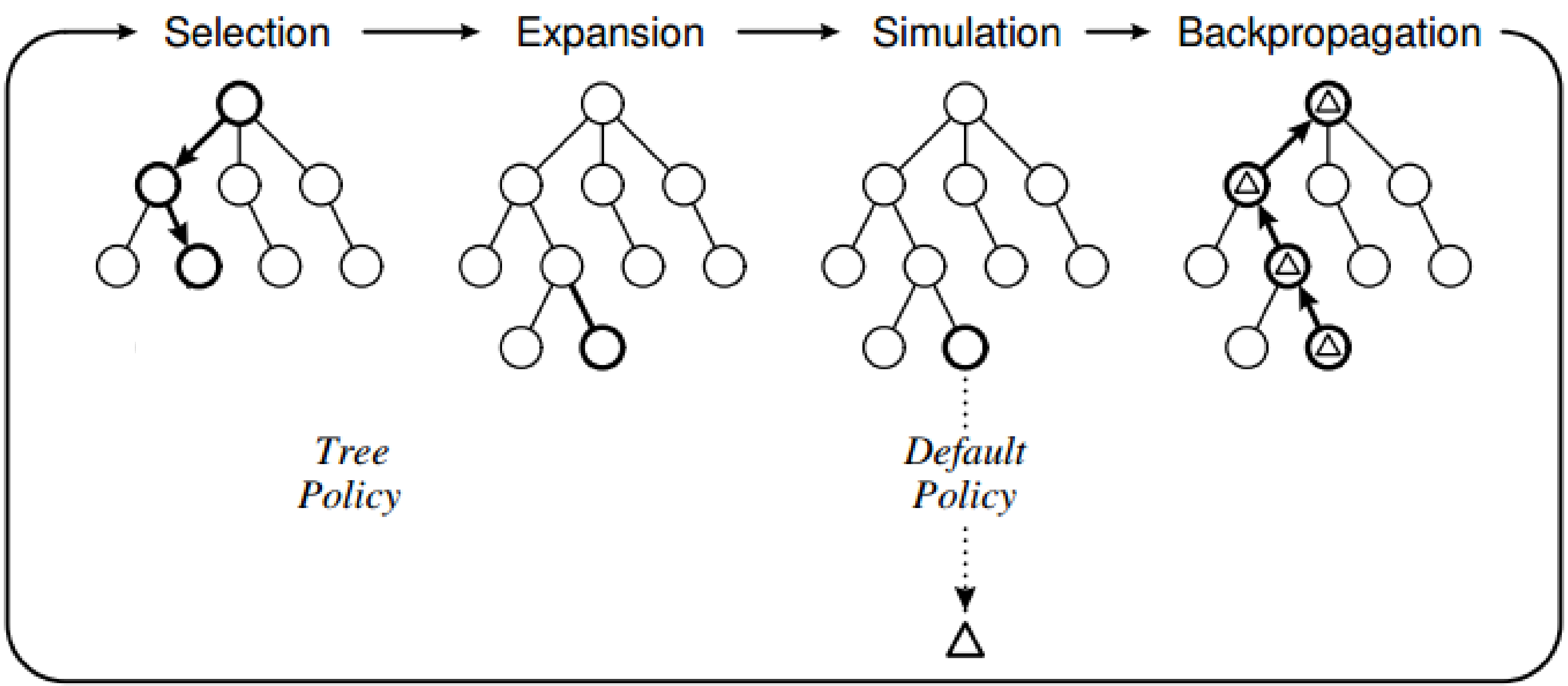
Monte Carlo Tree Search

- MCTS for single player setting: online planning in an unknown environment
- You are now in some state, need to choose an action, but you know nothing about the environment
- Helper: a simulator tells you your available actions, and reward after you take the action

Green player controlled by you
Actions={up, down, nothing}

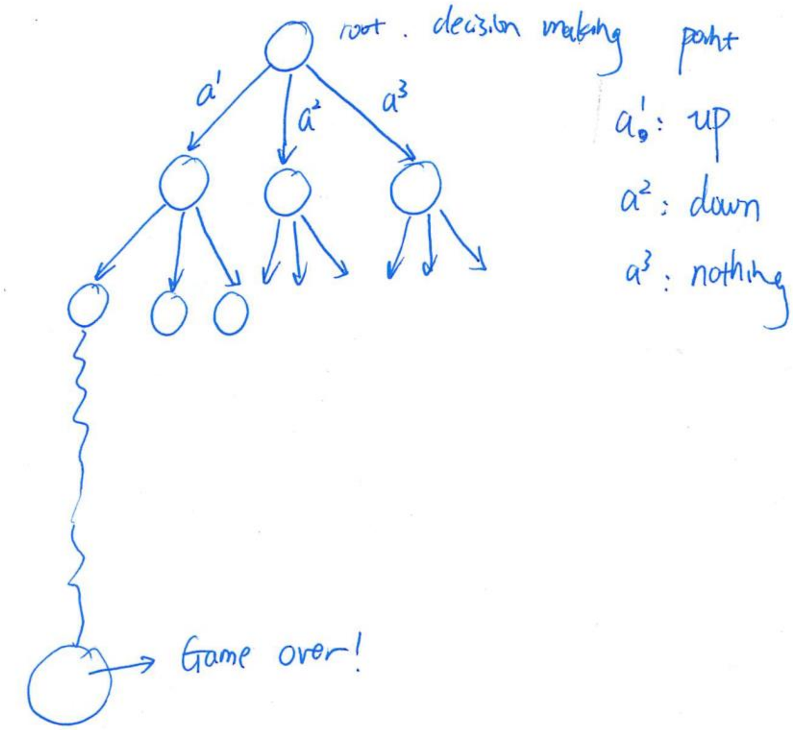
Monte Carlo Tree Search

- Build a search tree node by node
  - Node: state; Edge: available actions
- Repeat: Select→Expand→Simulate→Backpropagate



Monte Carlo Tree Search

- Build a search tree node by node
  - Node: state; Edge: available actions
- Repeat: Select→Expand→Simulate→Backpropagate

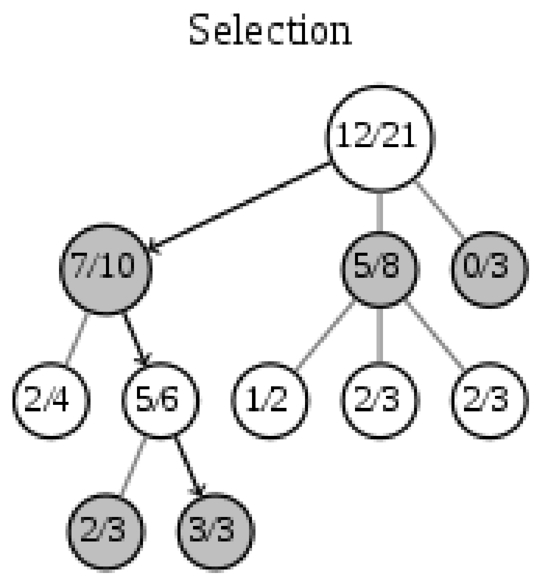


Monte Carlo Tree Search

- Simplest MCTS
  - In each iteration
    - Select: Choose the branch with the highest value
    - Expand: Add one node by randomly selecting an action
    - Simulate: Uniform random rollout
    - Backpropagate: update mean return (average accumulated reward) along the path
  - Output: action correspond to branch with highest value at the root node after K iterations

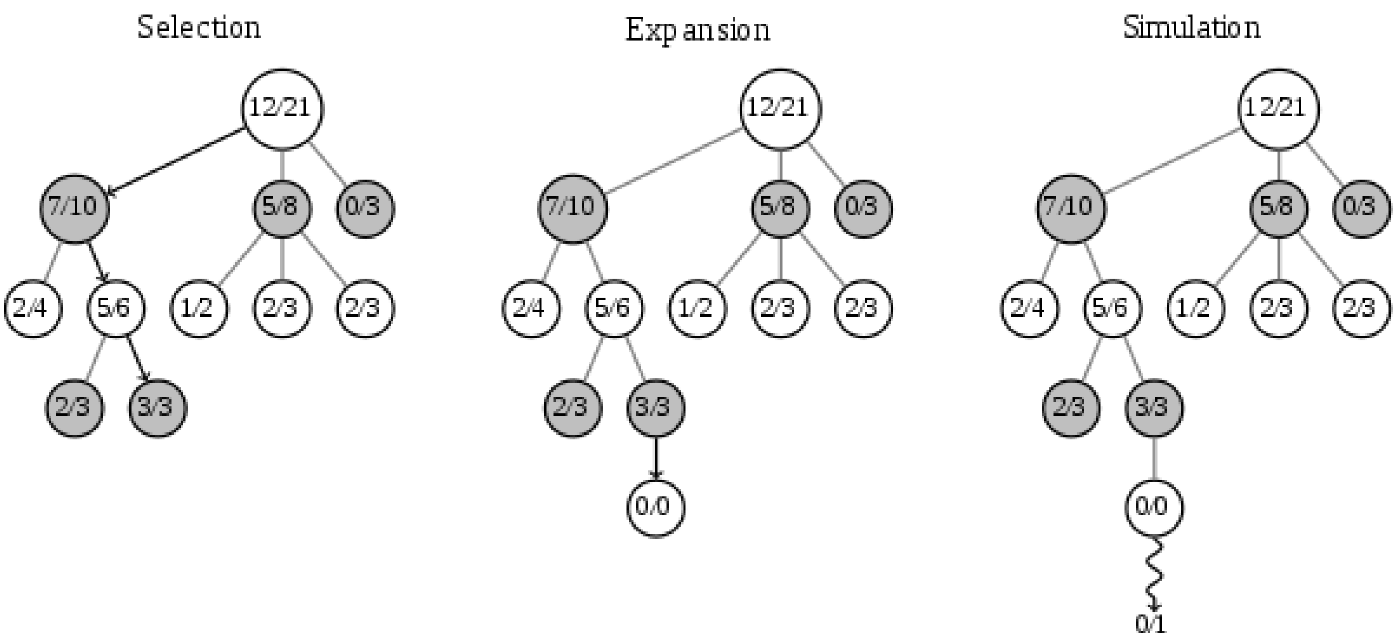
Monte Carlo Tree Search Example

Q: Assume the numbers in the nodes represent the mean return, which leaf node will be expanded when using the simplest MCTS?



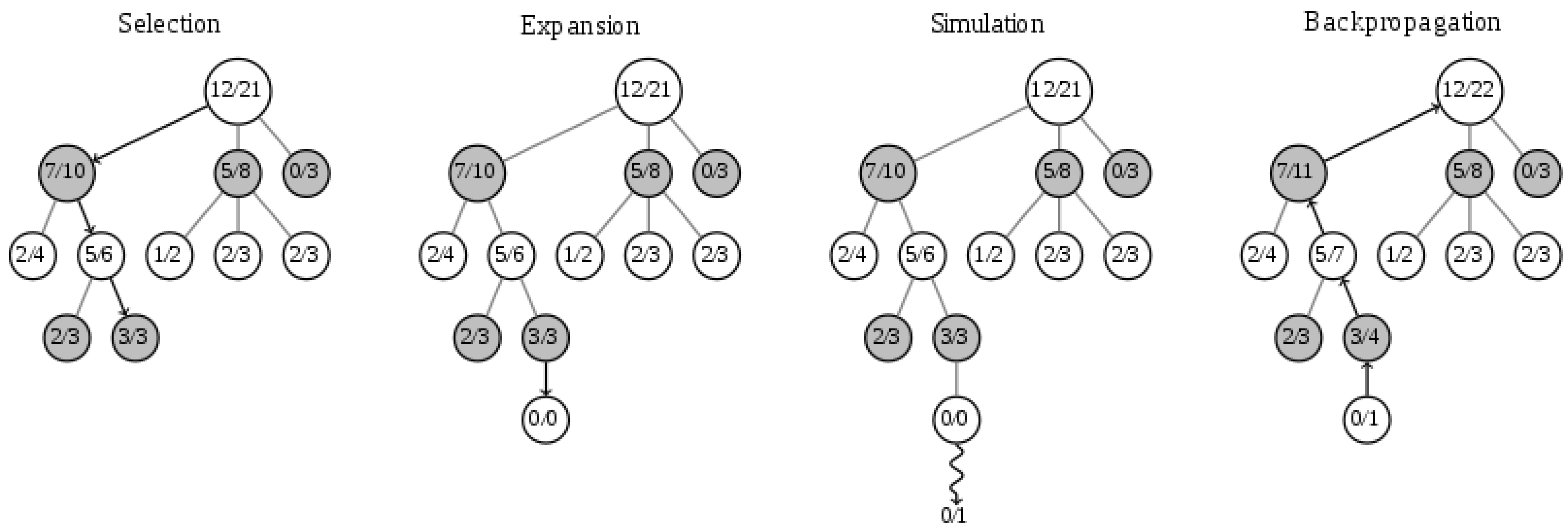
Monte Carlo Tree Search Example

Q: Assume the following tree is built for the Atari game, the numbers in the nodes represent the total number of times we win / the total number of times we visit the state, which nodes will be updated in the backpropagation step?



Monte Carlo Tree Search Example

Q: Assume the following tree is built for the Atari game, the numbers in the nodes represent the total number of times we win / the total number of times we visit the state, which nodes will be updated in the backpropagation step?



Recap: Upper Confidence Bound in MAB

- UCB1 Algorithm:
  - Always choose the arm with the highest upper confidence bound defined as $\mu_{UB}^{k} = \widehat{\mu_k} + \sqrt{\frac{2 \ln t}{N(k)}}$
  - Intuition: If μ_{UB}^{k} is large, either arm k is a good arm or $N(k)$ is small (not enough data is gathered)
  - General principle: optimism in the face of uncertainty

Monte Carlo Tree Search

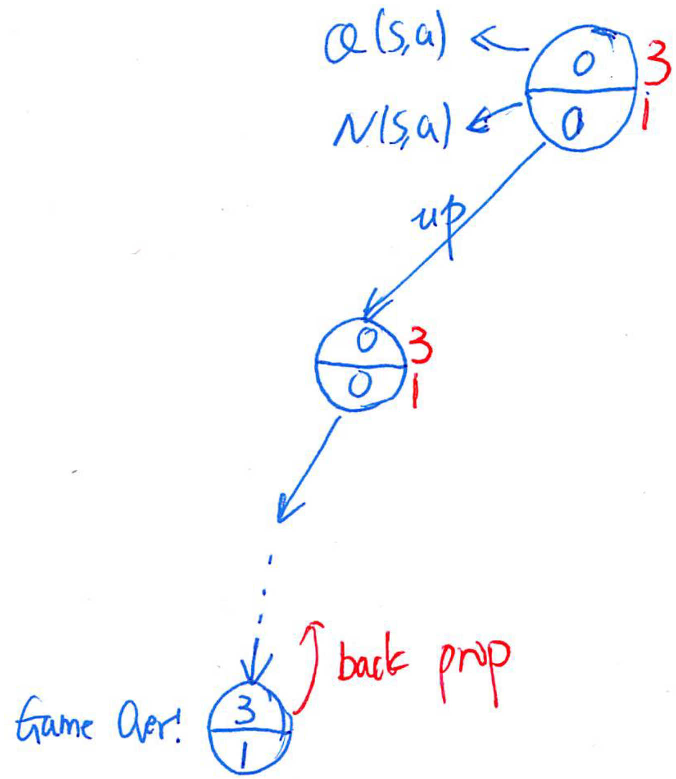
- Upper Confidence Bounds for Trees (UCT)
  - For each node, keep track of estimated action value and visit count: $Q(s,a)$ and $N(s,a)$
  - Select: Balance exploration vs exploitation:
    - If some actions never been chosen, randomly choose among them
    - Choose branch with highest Upper Confidence Bounds (UCB):

$$Q^{\oplus}(s,a) = Q(s,a) + c\sqrt{\frac{\ln N(s)}{N(s,a)}}$$

Monte Carlo Tree Search

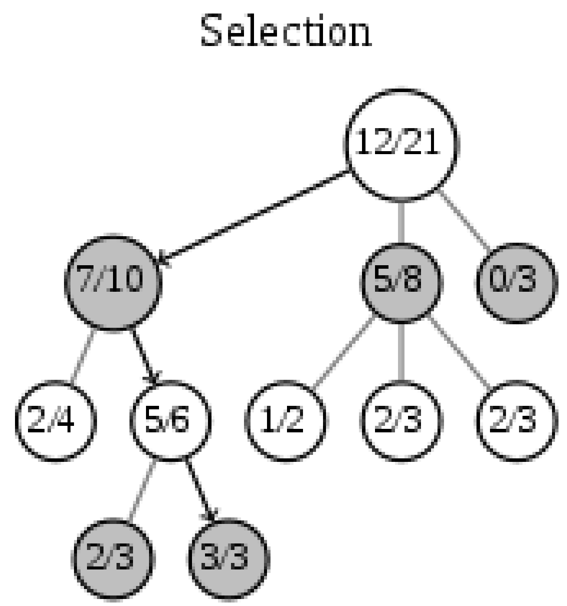
- Upper Confidence Bounds for Trees (UCT)
  - For each node, keep track of estimated action value and visit count: $Q(s,a)$ and $N(s,a)$
  - Select: Balance exploration vs exploitation:
    - If some actions never been chosen, randomly choose among them
    - Choose branch with highest Upper Confidence Bounds (UCB):

$$Q^{\oplus}(s,a) = Q(s,a) + c\sqrt{\frac{\ln N(s)}{N(s,a)}}$$



Monte Carlo Tree Search Example

Q: Assume the numbers in the nodes represent the Q(s,a) / N(s,a), which leaf node will be expanded when using UCT with $c = 10000$?



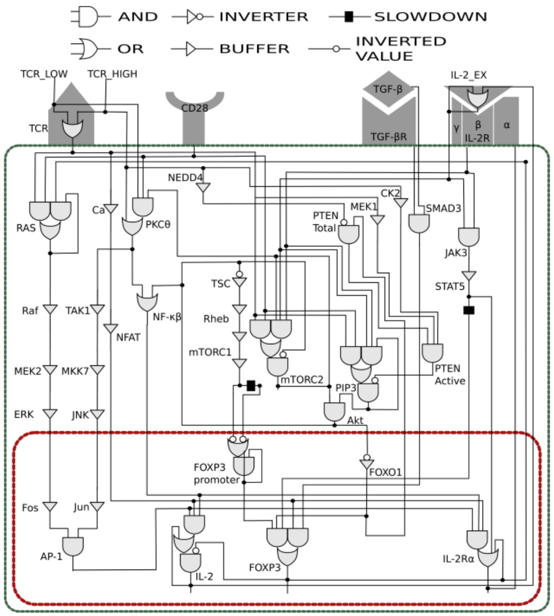
$$Q^{\oplus}(s,a) = Q(s,a) + c\sqrt{\frac{\ln N(s)}{N(s,a)}}$$

Monte Carlo Tree Search

- More advanced MCTS
  - Other advanced options:
    - Simulate: Terminate after T_0 steps and estimate the reward
    - Expand: Add more nodes to the tree
    - Output: Optimal action at root node, as well as Q and N in the subtree corresponds to the optimal action
    - Initialize search tree with domain knowledge

Example Application

- Steering immune system adaptation
  - Use MTCS to design a treatment plan, which steer T cell differentiation towards regulatory T cells or effector T cells
  - Have access to a leading T cell simulator
  - State: assignment of Boolean values to all the variables
  - Example action: activating or inhibiting cytokines, e.g., CD28, stimulating the T cell receptor



Resources

- Field Study in Deploying Restless Multi-Armed Bandits: Assisting Non-Profits in Improving Maternal and Child Health
- Sequential Planning for Steering Immune System Adaptation

Backup Slides

MCTS for Two-Player Complete Information Games

- Existing code package: mctspy (limited functionality)

For Tic-Tac-Toe
With mctspy 0.1.1

```
import numpy as np
from mctspy.tree.nodes import TwoPlayersGameMonteCarloTreeSearchNode
from mctspy.tree.search import MonteCarloTreeSearch
from mctspy.games.examples.tictactoe import TicTacToeGameState

state = np.zeros((3,3))
initial_board_state = TicTacToeGameState(state = state, next_to_move=1)

root = TwoPlayersGameMonteCarloTreeSearchNode(state = initial_board_state)
mcts = MonteCarloTreeSearch(root)
best_node = mcts.best_action(10000)
```


https://pypi.org/project/mctspy/#description

Gaussian Mixture Models

- Often used for clustering
- Group scene images into three groups, check average pixel intensity of Blue in each group

Gaussian Mixture Models

- Gaussian (Normal) distribution: most commonly used
  - Mean
  - Variance
  - Gaussian distribution: $X \sim \mathcal{N}(\mu, \sigma^2)$
  - $f(x|\mu, \sigma^2) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$

Gaussian Mixture Models

- Model: Mixture of Gaussian
  - Distribution of each class/group/cluster: Close to Gaussian
  - Aggregated distribution: Combination of weighted Gaussians
  - Weight: Relative proportion of each class

Gaussian Mixture Models

- Predict output given a hard-coded GMM
  - Input → Check probability for each group → Output
  - Check probability: Bayes' Theorem

$$P(A \mid B) = \frac{P(B|A)P(A)}{P(B)}$$

- Learn/Train GMM
  - Given a set of data
  - Find the weight, mean and variance of each Gaussian component (Expectation Maximization algorithm)
  - Code packages: sklearn (Python), fitgmdist (MATLAB)

Poll 1: Gaussian Mixture Models

- Given a GMM for a corpus of images as follows, with the average blue intensity as the only feature used
  - 30% images belong group 1: $\mathcal{N}(0.2, 0.1)$
  - x% images belong to group 2: $\mathcal{N}(0.8, 0.05)$
  - y% images belong to group 3: $\mathcal{N}(0.5, 0.1)$
  - x and y are unknown
  - For an image whose average blue intensity is 0.7, which group does it belong to?
    - Group 1
    - Group 2
    - Group 3
    - Cannot be determined