UiO : Department of informatics
University of Oslo

Adaptive Methods for Data-based Decision Making *Lecture 8*

IN-STK 5000 / 9000

Autumn 2022

Slides by Dr. Anne-Marie George, UiO

UNIVERSITY OF OSLO

dSience – Centre for Computational and Data Science

Menu

← News and events ← dScience events

Today!

Norwegian version of this page

dScience events

News

Data Science Day @ UiO 2022

dScience would like to welcome the Data Science community to the fifth annual Data Science Day.

Time and place: Oct. 19, 2022 5:00 PM–10:00 PM, The Science Library and Sophus Lie's auditorium

Add to calendar

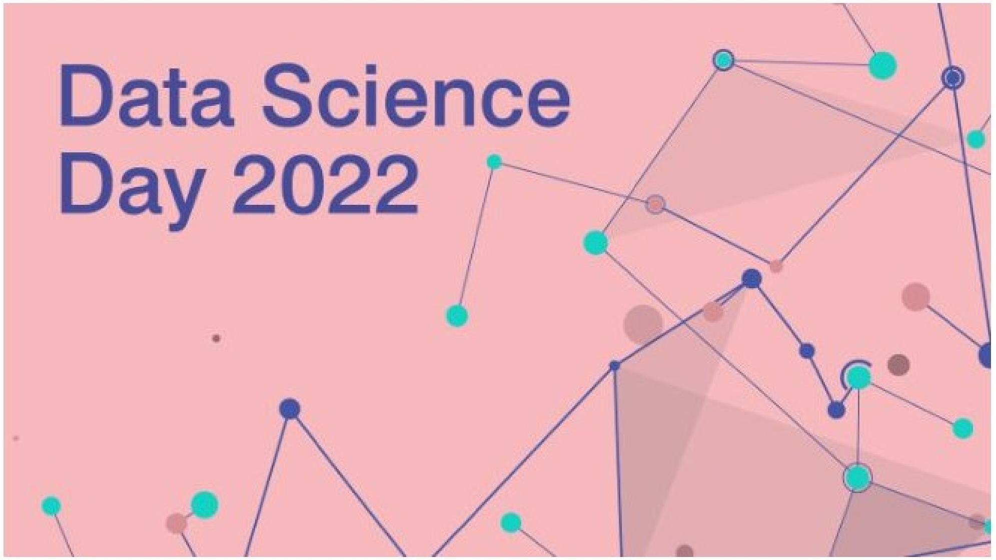


Illustration: Colourbox / AstroMaria

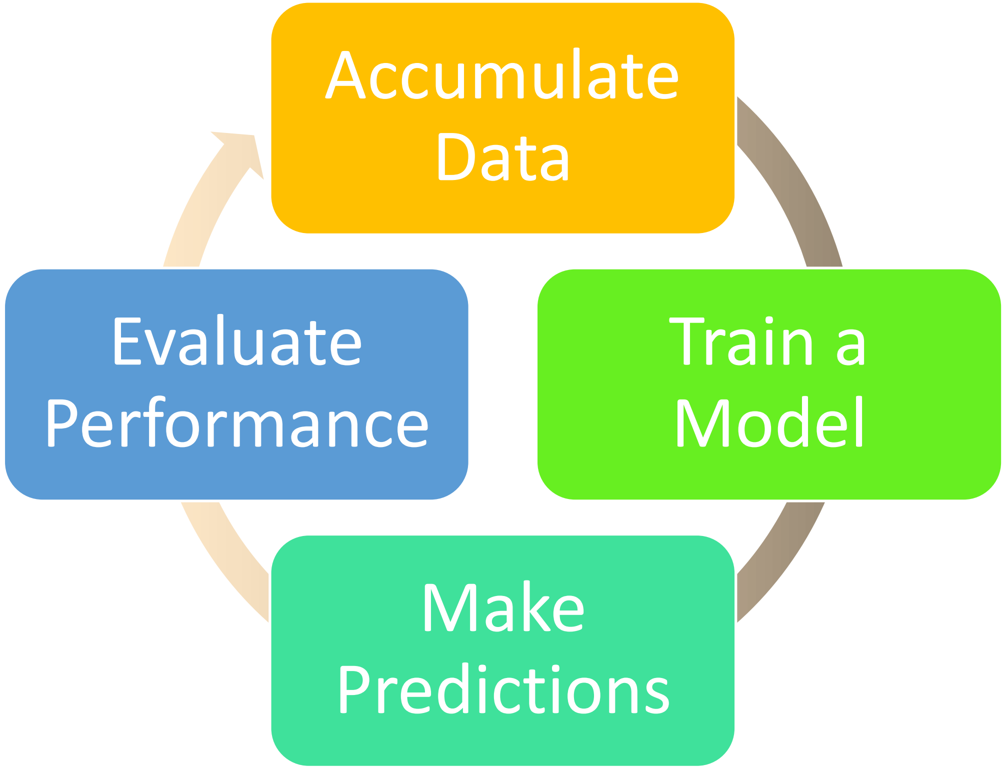
What we talk about today: Online Machine Learning

- Online Learning Settings
- Concept Drifts
- “Traditional” Online Learning: Regression & Clustering Examples
- Multi-Armed Bandits
- Total Rewards and Regret
- Bandit Policies

Online Learning Vs. Offline Learning

Sources: Wikipedia, River tutorial, …
Recommendations: Awesome online machine learning (github)

Sequentially update ML model as more data becomes available!



Learn from one *batch* of data (use complete dataset in one go).
→ Problem:

- Data might not fit in memory
- Data only available over time

Data Streams

Reactive Data Streams:

- Receive unfiltered live data.
  E.g., clicks on website, heart rate measures, …
- No influence over observations!

Proactive Data Streams:

- Control over the data stream.
  (Timing, order, etc. of observations)
  E.g., read data from file in specific order.

→ Turn reactive streams into proactive:
Save database and process offline.

→ <u>Challenge</u>:
Model trained offline (on proactive data) should perform correctly on reactive data.

Online Learning - Advantages

- Handles streams or updates of data → Adaptive to changes!
- Applications:
  - Recommender systems,
  - Anomaly detection,
  - Finance market, …
- Learn from one data point at a time:
  - No need to train a new model from scratch
  - No need to store all historic data
- Can be applied for cases where one-shot learning is not feasible due to abundance of data (*out-of-core learning*)

Online Learning - Challenges

- Monitoring for changes and continued retraining
  → How often necessary?
- Reduced performance compared to offline learning on complete data (if the distribution is static).
- Evaluation: ~~Cross-validation~~ Data must be in realistic order.

Concept Drifts

Data X (and labels y) is drawn from a probability distr. P

- Supervised learning: Learn function $f(x) = y$ that predicts labels

Concept drift ≈ Distribution P changes over time

- *Virtual* concept drift: $P(X)$ changes, while f remains unchanged.
- *Real* concept drift: $P(X, y)$ changes, i.e., f changes!
  - Abrupt change: Concept changes abruptly at given time.
  - Gradual change: Gradual concept change over time steps.
- Example: Energy consumption over year, traffic over week, …
- Unsupervised learning: Learn clusters, patterns, latent features, …
  → only virtual concept drifts are relevant

Drift Detector

- Offline learning performs badly under concept drifts
- Online learning updates model based on new data and can adapt to new concepts

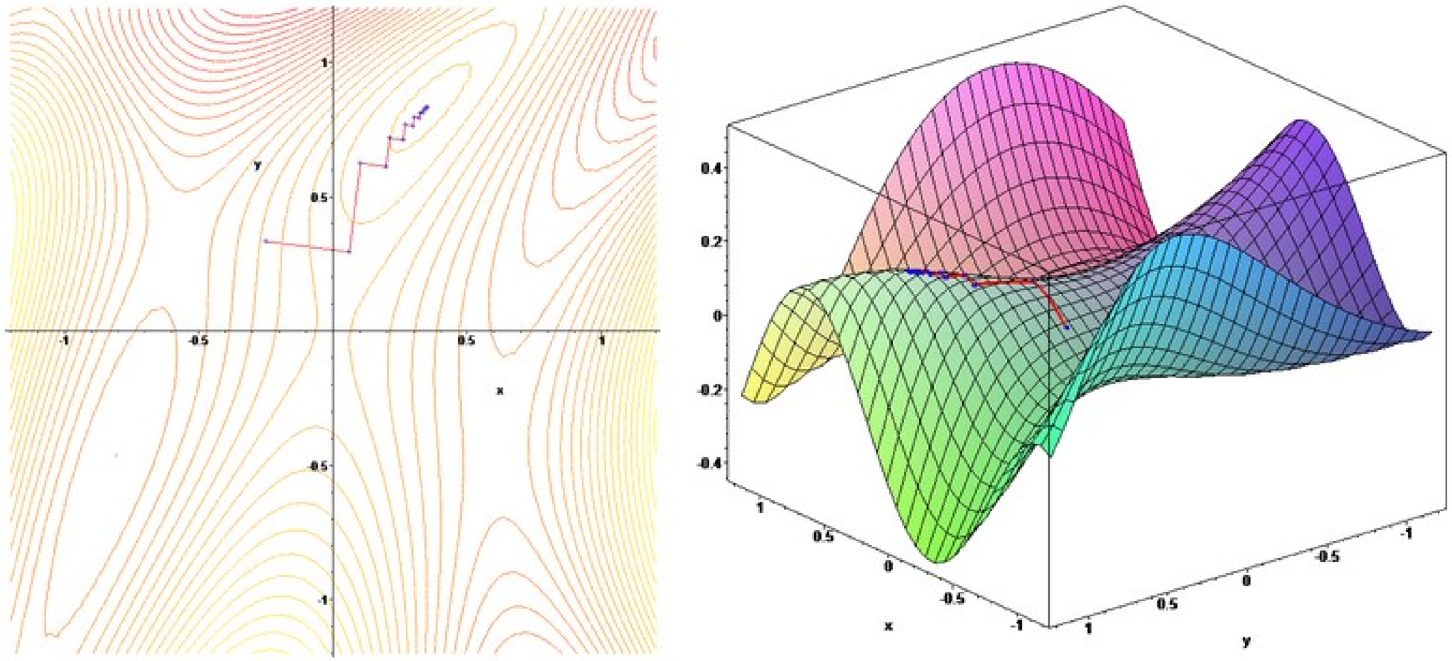
→Trigger model updates when concept drifts occur!

Drift-aware methods:

- Employ change detection mechanism ≈ drift detector
- Monitor model performance based on some metric
  → trigger model update when performance worsens

Online Regression via Stochastic Gradient Descent

- <u>Gradient Descent</u>: Find a local minimum of a function F.
  → Start from a random point, then repeatedly "take a step" in the direction of the steepest descent = $-\nabla F$



See [Wikipedia on Gradient Descent]

Online Regression via Stochastic Gradient Descent

- <u>Gradient Descent</u>: Find a local minimum of a function F.
  → Start from a random point, then repeatedly "take a step" in the direction of the steepest descent = $-\nabla F$
- <u>GD for Regression</u>: Min. prediction error $F(x, w)$ (e.g. MAE, MSE) for regression function with parameters w over complete data set x!
  Update: $w_{n+1} = w_n - \gamma \nabla F(x, w_n),\ n \geq 0.$
- <u>Stochastic GD</u>: Update parameters w sequentially for each data point individually $x_1,\ x_2, \ldots$:
  $$w_{n+1} = w_n - \gamma \nabla F(x_n, w_n),\ n \geq 0.$$
  → Can be updated as new data becomes available!

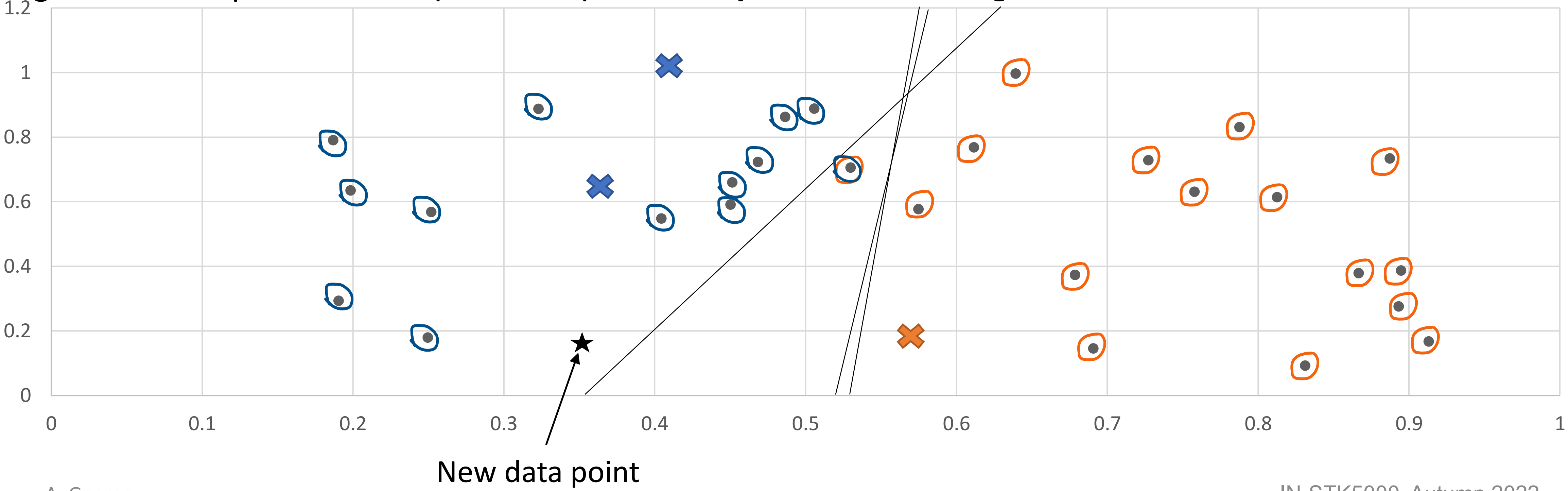
Online Clustering with k-Means

<u>New data point</u> (x, y):
- Allocate new point to a cluster (by nearest cluster center).
- Shift cluster center according to new point.

0. Insert k random cluster *centroids* **(e.g. on k data points)**

1. Repeat: Cluster Assignment + Move Centroid **to cluster average**

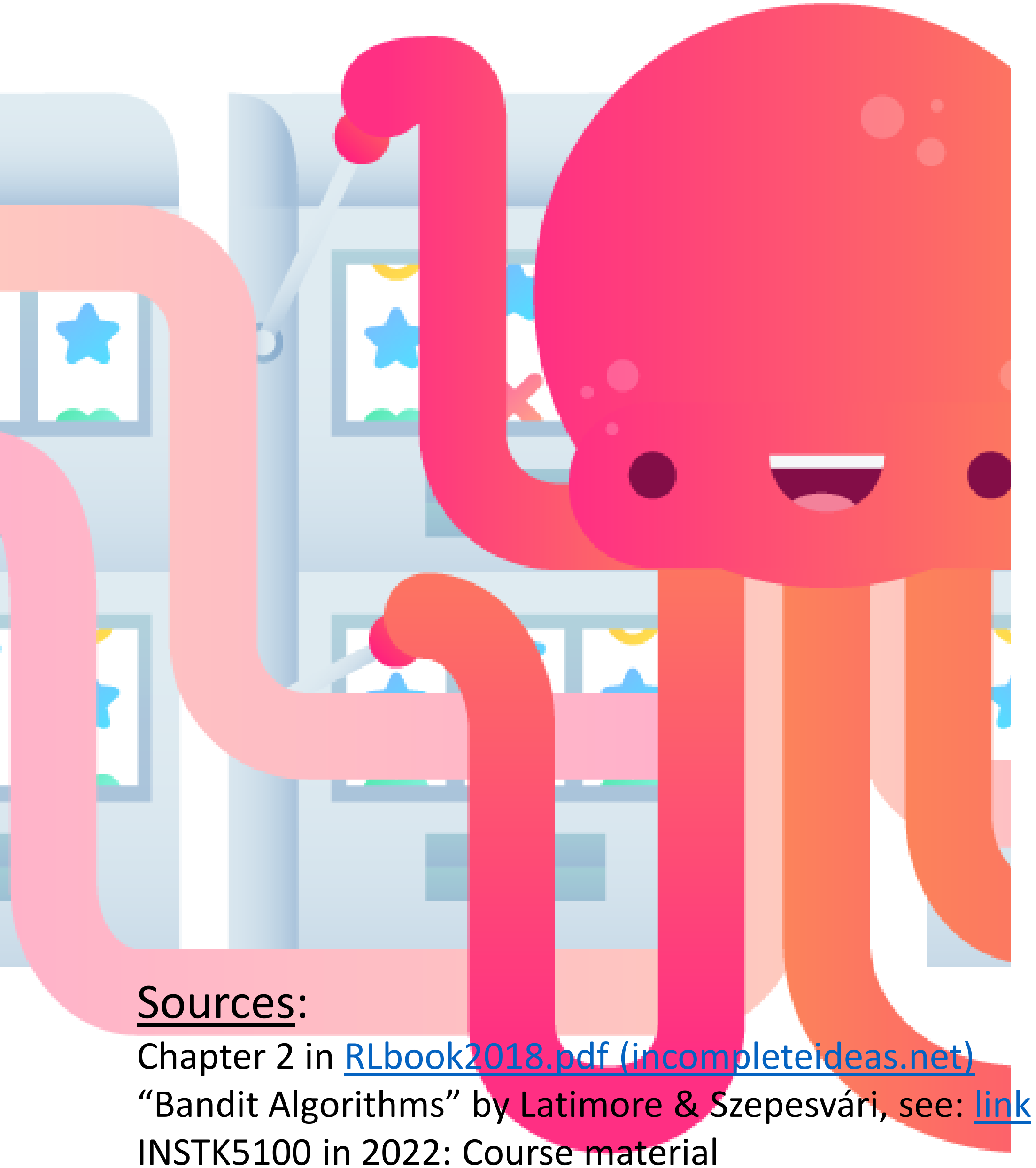
Algorithm Example: k-Means (for $k = 2$)



River

[online-ml/river: Online machine learning in Python (github.com)]

- Python Library for *Online Machine Learning*
- Merger between `scikit-multiflow` and `creme`
- Includes:
  - Algorithms (for classification, regression, clustering, bandits)
  - data-transformation methods,
  - drift detectors,
  - datasets,
  - performance metrics



Another Online Problem:

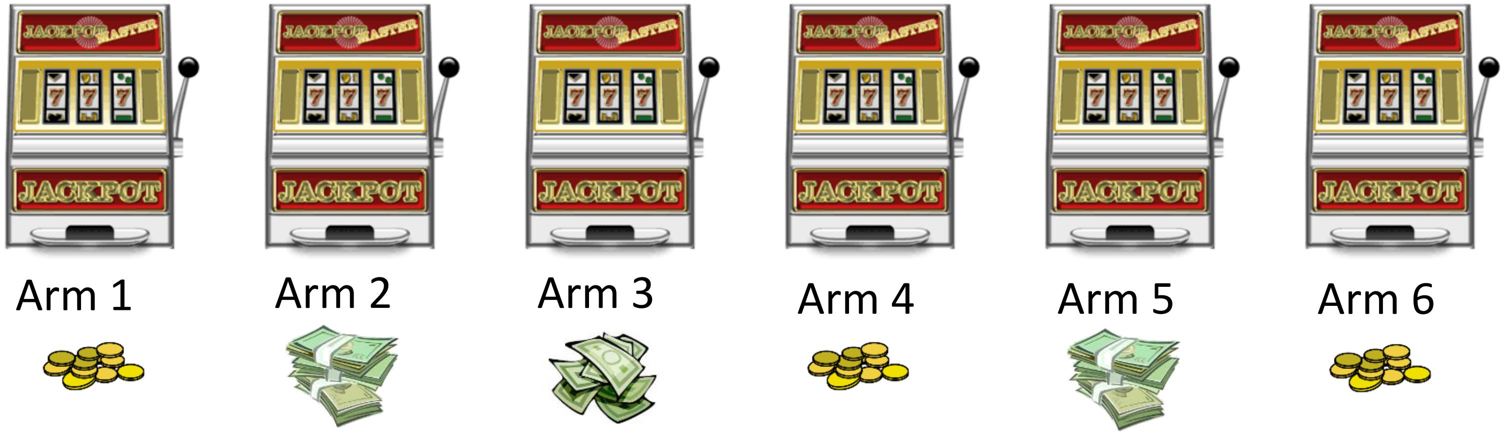
Multi-armed Bandits

- Online problem:
  At every step choose an action
- Feedback:
  (numeric) reward for action
  → Proactive Data Stream

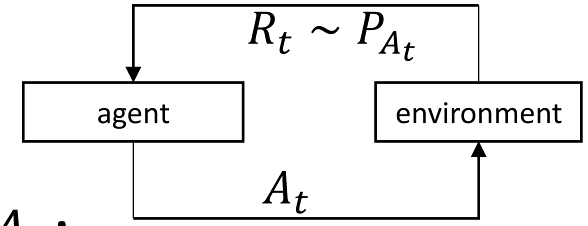
Sources:
Chapter 2 in RLbook2018.pdf (incompleteideas.net)
"Bandit Algorithms" by Latimore & Szepesvári, see: link
INSTK5100 in 2022: Course material

Multi-armed Bandits: Setting

- The Bandits:



- Actions: At any time step choose one arm to pull.
- Loop: Select action A_t, observe feedback (reward) R_t from unknown distribution P_{A_t}.



- Goal: Maximise rewards over time.

The Exploration Exploitation Trade-off

- Exploration: Try out new actions that might turn out to be beneficial (but might not).
- Exploitation: Play actions that have been played in the past and turned out to be good.

→ Only exploring or only exploiting is suboptimal (prevents us from achieving high rewards)
→ We need to find an appropriate tradeoff between exploring and exploiting!

?	Wins: 4 Losses: 3	Wins: 6 Losses: 1	?	?	?

Which arm should I pull next?
Should I try a new one?

Note: Exploration-Exploitation Dilemma is not an issue for un-/supervised learning problems

$\boldsymbol{k}$-Bandits Problem: Example

- Trying out Restaurants/Bars in Oslo:
- Not every visit (to the same place) is equally good/bad
- Want to sequentially choose a place to go:
  - Select promising places!

  → Need to learn which places are good!

$\boldsymbol{k}$-Bandits Problem: Example

- <u>Medical treatments</u>:
- Suppose there are several treatments available
- The treatments have unknown success rates
- Want to sequentially prescribe treatments to patients:
  - Give promising treatments to patients!

  → Need to learn which treatment is most effective!

At time t:
Action $A_t \in [k]$
Reward $R_t \sim P_{A_t}$
Selection prob. $P_{strategy}(a,t)$
Est. val. $\mathrm{Q}^{\mathrm{t}}(a)$
Real val. $q^*(a) = \mathbb{E}[R_t | A_t = a]$

Estimating Action Values

- The **value of an action** is its expected reward:
$$\boldsymbol{q}^*(\boldsymbol{a}) = \mathbb{E}[R_t | A_t = a] = \int_{r \in R} r \, \mathrm{d}P_a(r).$$
→ Reward distribution is unknown, thus the values of actions are also unknown!
- Idea: Estimate the values of actions based on prior feedback!

  The **estimated value of an action** at time t: $\boldsymbol{Q}^{\boldsymbol{t}}(\boldsymbol{a})$
  → We want $\mathrm{Q}^{\mathrm{t}}(a)$ to be close to $q^*(a)$.
- Sample-average:
$$Q^t(a) = \frac{\sum \textit{rewards when a was played}}{\# \ \textit{a was played in prior rounds}} = \frac{\sum_{i=1}^{t-1} R_i \cdot \mathbb{I}_{A_i = a}}{\sum_{i=1}^{t-1} \mathbb{I}_{A_i = a}} \qquad \text{if } \sum_{i=1}^{t-1} \mathbb{I}_{A_i = a} \neq 0$$
$$Q^t(a) = 0 \text{ (or any other constant)} \qquad \text{otherwise}$$

Expected Total Reward

<u>At time t</u>:
Action $A_t \in [k]$
Reward $R_t \sim P_{A_t}$
Selection prob. $P_{strategy}(a, t)$
Est. val. $\mathrm{Q}^{\mathrm{t}}(a)$
<u>Real val.</u> $q^*(a) = \mathbb{E}[R_t | A_t = a]$

- Let $P_{strategy}(a, t)$ denote the probability of pulling arm a in round t according to a fixed strategy.
- The **expected total reward** achieved by the strategy over T rounds is:

$$\mathbb{E}[\textstyle\sum_{t=0\ldots T-1} R_t]$$
$$= \sum_{t=0\ldots T-1} \mathbb{E}[R_t] = \sum_{t=0\ldots T-1} \sum_{a\in[k]} P_{strategy}(a, t) \cdot \mathbb{E}[R_t | A_t = a] = \sum_{t=0\ldots T-1} \sum_{a\in[k]} P_{strategy}(a, t) \cdot q^*(a)$$

Quality Measure: Regret

At time t:
Action $A_t \in [k]$
Reward $R_t \sim P_{A_t}$
Selection prob. $P_{strategy}(a,t)$
Est. val. $\mathrm{Q}^{\mathrm{t}}(a)$
Real val. $q^*(a) = \mathbb{E}[R_t | A_t = a]$
Best val. q^{max}, best arm a^*

- The **expected total reward** achieved by the strategy over T rounds is:

$$\mathbb{E}\left[\sum_{t=0\dots T-1} R_t\right] = \sum_{t=0\dots T-1} \sum_{a\in[k]} P_{strategy}(a,t) \cdot q^*(a)$$

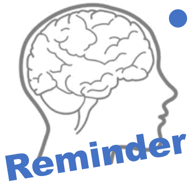
- Intuition: Find a strategy that is as close as possible to always pulling the "best" arm a^* with value q^{max}
- The **regret** of a strategy is

$$\begin{aligned} \mathrm{regret_T(strategy)} &= \mathrm{T} \cdot q^{max} - \mathbb{E}[\textstyle\sum_{t=0\dots T-1} R_t] \\ = \ \dots &= \textstyle\sum_{a\in[k]}[q^{max} - q^*(a)] \cdot \mathbb{E}[\# \ a \ is \ pulled] \end{aligned}$$

  - The regret is always ≥ 0.
  - The regret of a strategy that selects only best action (actions with maximal value) is 0.
  - Ideally, we play a strategy that gives us a regret that is sub-linear in T.
  - There is a known lower bound on the regret: $O(\sqrt{T \cdot k})$

The Exploration Exploitation Trade-off

- Exploration: Try out new actions that might turn out to be beneficial (but might not).
- Exploitation: Play actions that have been played in the past and turned out to be good.



→ Only exploring or only exploiting is suboptimal (prevents us from achieving high rewards)
→ We need to find an appropriate tradeoff between exploring and exploiting!

Exploration: Trying out (possibly random) actions.
→ Helps us getting better estimates of the action values $Q^t(a)$ and to identify the best action for future turns.

Exploitation: Playing a greedy action, i.e.,
one with currently highest estimated value $argmax_{a \in [k]} Q^t(a)$ (always exists!).
→ Gives us highest expected immediate reward w.r.t to our current estimates.

Let’s take a Quiz…

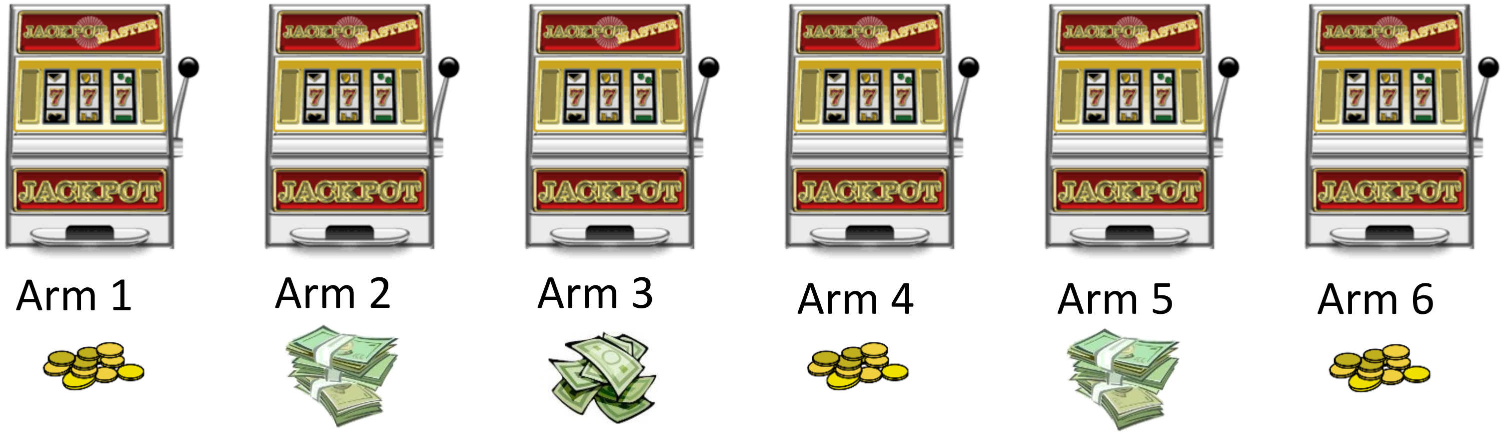
… go to Mentimeter!

Let’s take a break…

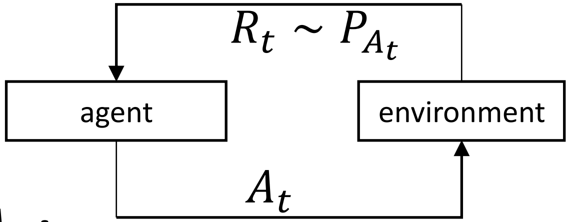
Back on in 5 min!

Multi-armed Bandits: Setting

- The Bandits:



- Actions: At any time step choose one arm to pull.
- Loop: Select action A_t, observe feedback (reward) R_t from unknown distribution P_{A_t}.



- Goal: Maximise rewards over time (e.g. min. prediction error).

A Simple Strategy: Explore-Then-Commit (ETC)

Simulating 3 arms with $\boldsymbol{m = 2}$

For $\mathrm{t} < m \cdot k$:

- Choose action $A_{t \bmod k}$
- Observe reward R_t

For $t \geq m \cdot k$:

- Compute
  $$Q^{m \cdot k-1}(a) = \frac{\sum_{i=0}^{m \cdot k-1} R_i \cdot \mathbb{I}_{A_i=a}}{m}$$
- Select arm
  $$A_t \in argmax_{a \in [k]} Q^{k \cdot m-1}(a)$$

Round	Arm 1 R_t | N^{t+1} | Q^{t+1}	Arm 2 R_t | N^{t+1} | Q^{t+1}	Arm 3 R_t | N^{t+1} | Q^{t+1}
0	0 | 1		
1		0 | 1	
2			1 | 1
3	1 | 2 | 1/2		
4		1 | 2 | 1/2	
5			0 | 2 | 1/2
6			
7		???	
8			
9			
10			

A Simple Strategy: Explore-Then-Commit (ETC)

Simulating 3 arms with $\boldsymbol{m = 3}$

For t $< m \cdot k$:

- Choose action $A_{t \bmod k}$
- Observe reward R_t

For $t \geq m \cdot k$:

- Compute
  $$Q^{m \cdot k-1}(a) = \frac{\sum_{i=0}^{m \cdot k-1} R_i \cdot \mathbb{I}_{A_i=a}}{m}$$
- Select arm
  $$A_t \in argmax_{a \in [k]} Q^{k \cdot m-1}(a)$$

Round	Arm 1			Arm 2			Arm 3		
	R_t	N^{t+1}	Q^{t+1}	R_t	N^{t+1}	Q^{t+1}	R_t	N^{t+1}	Q^{t+1}
0	0	1							
1				0	1				
2							1	1	
3	1	2							
4				1	2				
5							0	2	
6	0	3	1/3						
7				1	3	2/3			
8							0	3	1/3
9				0	3	2/4			
10							…		

Explore-Then-Commit (ETC): Regret

<u>At time t</u>:
Action $A_t \in [k]$
Reward $R_t \sim P_{A_t}$
Selection prob. $P_{strategy}(a,t)$
Est. val. $\mathrm{Q}^{\mathrm{t}}(a)$
<u>Real val.</u> $q^*(a) = \mathbb{E}[R_t | A_t = a]$
Best val. q^{max}, best arm a^*

- The regret of ETC after T rounds is

$$\mathrm{regret}_{\mathrm{T}}(\mathrm{ETC}) \leq \sum_{a \in [k]} [q^{max} - q^*(a)] \cdot m + \sum_{a \in [k]} [q^{max} - q^*(a)] \cdot (T - m \cdot \mathrm{k}) \cdot \exp\left(-\frac{m\left(q^{max} - q^*(a)\right)^2}{4}\right)$$

- Problem dependent regret bound:
  - Depends on the specific instance (because it includes the terms $q^{max} - q^*(a)$).
- Exploration-Exploitation:
  - If m is large, i.e., we explore a lot, then the first term gets large.
  - If m is small, i.e., we concentrate on exploitation, then the second term gets large.
- Linear in T: We can't get lower average regret by increasing the number of rounds (after $T > m \cdot k$).

$\boldsymbol{\epsilon}$-Greedy Action Selection

At time t:
Action $A_t \in [k]$
Reward $R_t \sim P_{A_t}$
Selection prob. $P_{strategy}(a, t)$
Est. val. $\mathrm{Q}^{\mathrm{t}}(a)$
Real val. $q^*(a) = \mathbb{E}[R_t | A_t = a]$
Best val. q^{max}, best arm a^*

- Exploit:
  Most of the time.
- Explore:
  Instead of a fixed explore phase, just sometimes (randomly) choose to explore.

  → Exploration chance: $\epsilon \in [0,1]$

ϵ-Greedy Action Selection

At time t:
Action $A_t \in [k]$
Reward $R_t \sim P_{A_t}$
Selection prob. $P_{strategy}(a, t)$
Est. val. $\mathrm{Q}^t(a)$
#pulls $N^t(a)$
Real val. $q^*(a) = \mathbb{E}[R_t | A_t = a]$
Best val. q^{max}, best arm a^*

- Let $\epsilon \in [0,1]$.
- Initialise $\mathrm{Q}^0(a) = const.$ for all $a \in [k]$.
- Initialise $N^0(a) = 0$ for all $a \in [k]$, number times each arm was pulled.
- For $t = 0 \dots T - 1$:
    - With probability $(1 - \epsilon)$: $A_t = argmax_{a \in [k]} Q^t(a)$ # select a greedy action
    - With probability ϵ: $A_t \sim \mathrm{U}([\mathrm{k}])$ # sample uniformly rand. from $\{1, \dots, k\}$
  - Receive reward R_t
  - $N^{t+1}(A_t) \mathrel{+}= 1$ and $N^{t+1}(a) = N^t(a)$ for all other actions a.
  - $Q^{t+1}(a) = \begin{cases} Q^t(a) + \frac{1}{N^{t+1}(a)}[R_t - Q^t(a)] & if\ a = A_t \\ Q^t(a) & otherwise \end{cases}$

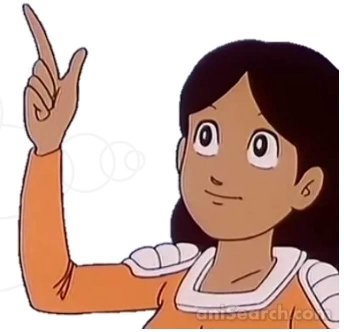
ϵ-Greedy Action Selection

<u>At time t</u>:
Action $A_t \in [k]$
Reward $R_t \sim P_{A_t}$
Selection prob. $P_{strategy}(a,t)$
Est. val. $\mathrm{Q}^{t}(a)$
#pulls $N^{t}(a)$
<u>Real val.</u> $q^*(a) = \mathbb{E}[R_t | A_t = a]$
Best val. q^{max}, best arm a^*

- Let $\epsilon \in [0,1]$.
- Initialise $\mathrm{Q}^0(a) = const.$ for all $a \in [k]$.
- Initialise $N^0(a) = 0$ for all $a \in [k]$, number times each arm was pulled.
- For $t = 0 \ldots T - 1$:
    - With probability $(1-\epsilon)$: $A_t = argmax_{a \in [k]} Q^t(a)$ # select a greedy action
    - With probability ϵ: $A_t \sim \mathrm{U}([\mathrm{k}])$ # sample uniformly rand. from $\{1, \ldots, k\}$
  - Receive reward R_t
  - $N^{t+1}(A_t) \mathrel{+}= 1$ and $N^{t+1}(a) = N^t(a)$ for all other actions a.
  - $Q^{t+1}(a) = \begin{cases} Q^t(a) + \frac{1}{\ldots}[R_t - Q^t(a)] & if\ a = A_t \end{cases}$

If $T \to \infty$ can we guarantee $Q^t(a) \to q^*(a)$ for all actions a?

<u>Yes</u>: Because for $T \to \infty$ we will select every arm infinitely often.



Greedy Action Selection: Example

Simulating 3 arms with $\boldsymbol{\epsilon = 0}$
→ <u>only greedy actions</u>

Initialise

- $\mathbf{Q^0}(\boldsymbol{a}) = \mathbf{0}$ for all $a \in [3]$
- $\boldsymbol{N^0}(\boldsymbol{a}) = \mathbf{0}$ for all $a \in [3]$

Repeat

- Choose greedy action A_t
- Observe reward R_t
- $N^{t+1}(A_t) \mathrel{+}= 1$
- $Q^{t+1}(A_t) = Q^t(A_t) + \frac{1}{N^{t+1}(A_t)}[R_t - Q^t(A_t)]$



Round	Arm 1			Arm 2			Arm 3		
	R_t	N^{t+1}	Q^{t+1}	R_t	N^{t+1}	Q^{t+1}	R_t	N^{t+1}	Q^{t+1}
0	0	1	0		0	0		0	0
1				0	1	0			
2							1	1	1
3							0	2	1/2
4							0	3	1/3
5							1	4	1/2
6							1	5	3/5
7							…		

Greedy Action Selection: Example

Simulating 3 arms with $\boldsymbol{\epsilon = 0}$
→ <u>only greedy actions</u>

Initialise

- $\mathbf{Q^0(a) = 2}$ for all $a \in [3]$
- $\boldsymbol{N^0(a) = 0}$ for all $a \in [3]$

Repeat

- Choose greedy action A_t
- Observe reward R_t
- $N^{t+1}(A_t) \mathrel{+}= 1$
- $Q^{t+1}(A_t) = Q^t(A_t) + \frac{1}{N^{t+1}(A_t)}[R_t - Q^t(A_t)]$

Round	Arm 1			Arm 2			Arm 3		
	R_t	N^{t+1}	Q^{t+1}	R_t	N^{t+1}	Q^{t+1}	R_t	N^{t+1}	Q^{t+1}
0	0	1	0		0	2		0	2
1				0	1	0			
2							1	1	1
3							0	2	1/2
4							0	3	1/3
5							1	4	1/2
6							1	5	3/5
7							…		

Greedy Action Selection: Example

Simulating 3 arms with $\epsilon = \mathbf{0}$
→ <u>only greedy actions</u>

Initialise

- $\mathbf{Q^0}(\boldsymbol{a}) = -\mathbf{1}$ for all $a \in [3]$
- $\boldsymbol{N^0}(\boldsymbol{a}) = \mathbf{0}$ for all $a \in [3]$

Repeat

- Choose greedy action A_t
- Observe reward R_t
- $N^{t+1}(A_t) \mathrel{+}= 1$
- $Q^{t+1}(A_t) = Q^t(A_t) + \frac{1}{N^{t+1}(A_t)}[R_t - Q^t(A_t)]$

Round	Arm 1 R_t	N^{t+1}	Q^{t+1}	Arm 2 R_t	N^{t+1}	Q^{t+1}	Arm 3 R_t	N^{t+1}	Q^{t+1}
0	<u>0</u>	1	0		0	-1		0	-1
1	<u>1</u>	2	1/2						
2	<u>0</u>	3	1/3						
3	<u>0</u>	4	1/4						
4	<u>0</u>	5	1/5						
5	<u>0</u>	6	1/6						
6	<u>0</u>	7	1/7						
7	…								

Greedy Action Selection: Example

Simulating 3 arms with $\boldsymbol{\epsilon = 0.5}$
→ random actions ~half the time

Initialise

- $\mathbf{Q^0(a) = 0}$ for all $a \in [3]$
- $\boldsymbol{N^0(a) = 0}$ for all $a \in [3]$

Repeat

- Choose action A_t (greedy/rand.)
- Observe reward R_t
- $N^{t+1}(A_t) \mathrel{+}= 1$
- $Q^{t+1}(A_t) = Q^t(A_t) + \frac{1}{N^{t+1}(A_t)}[R_t - Q^t(A_t)]$

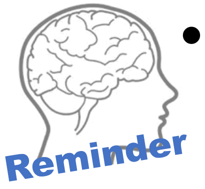


Round	Arm 1 R_t	N^{t+1}	Q^{t+1}	Arm 2 R_t	N^{t+1}	Q^{t+1}	Arm 3 R_t	N^{t+1}	Q^{t+1}	
0	0	1	0		0	0		0	0	←(random)
1				0	1	0				←(random)
2							1	1	1	←(random)
3							0	2	1/2	
4	1	2	1/2							←random
5	0	3	1/3							←(random)
6							0	3	1/3	
7				1	2	1/2				←random
8				1	3	2/3				
9				0	4	1/2				
10							1	4	1/2	←random
11							1	5	3/5	←(random)
12				1	5	3/5				←random
13										

$\boldsymbol{\epsilon}$-Greedy Action Selection: Summary

- Exploration rate can be tuned by varying the parameter $\epsilon \in [0,1]$.
- $\boldsymbol{\epsilon}$-Greedy has linear regret: $\text{regret}_{\mathrm{T}}(\epsilon - Greedy) \in \mathrm{O}(T)$.
- <u>Advantages</u>:
  - Easy to implement!
  - Vary the exploration parameter over time $\epsilon_t \in [0,1]$ such that, e.g., exploration rate decreases.
- <u>Issues</u>:
  - When ϵ-Greedy selects a non-greedy action, it does not differentiate between any of the actions (just picks uniformly at random)

Non-Stationary Rewards



- Reward of arm A_t follows *unknown* distribution $P(r_t | A_t = a) = P_{\theta_a}(r)$.

  Example:

  - Bernoulli distribution: P_{θ_a}, such that $r = \begin{cases} 1\ with\ prob. & \theta_a \\ 0\ with\ prob. & 1 - \theta_a \end{cases}$.
  - Normal distribution: P_{θ_a}, with mean $\theta_{a,\mu}$ and standard deviation $\theta_{a,\sigma}$.

- Now: Let us consider non-stationary rewards **i.e., a Concept Drift!**
  → The distribution of the rewards can change over time.
  → We want to give more recent rewards more weight than rewards from long ago timesteps…

$\boldsymbol{\epsilon}$-Greedy Action Selection with weighted averages

- Let $\epsilon \in [0,1]$.
- Initialise $\mathrm{Q}^0(a) = const.$ for all $a \in [k]$.
- Initialise $N^0(a) = 0$ for all $a \in [k]$, number times each arm was pulled.
- For $t = 0 \dots T-1$:
    - With probability $(1-\epsilon)$: $A_t = argmax_{a\in[k]} Q^t(a)$ # select a greedy action
    - With probability ϵ: $A_t \sim \mathrm{U}([\mathrm{k}])$ # sample uniformly rand. from $\{1, \dots, k\}$
  - Receive reward R_t
  - $N^{t+1}(A_t) \mathrel{+}= 1$ and $N^{t+1}(a) = N^t(a)$ for all other actions a.
  - $Q^{t+1}(a) = \begin{cases} Q^t(a) + \boldsymbol{\alpha^t(a)}[R_t - Q^t(a)] & if\ a = A_t \\ Q^t(a) & otherwise \end{cases}$

Weighted Averages

- Constant step size $\alpha \in (0,1]$:

$$Q^{t+1}(a) = Q^t(a) + \alpha \cdot [\mathrm{R_t} - \mathrm{Q^t(a)}] = \ \ldots$$
$$= (1-\alpha)^{N^{t+1}(a)} Q^0(a) + \sum_{i=1}^{t} \alpha(1-\alpha)^{N^{t+1}(a)-N^{i+1}(a)} R_i \mathbb{I}_{A_i=a}$$

- Non-Constant step size $\alpha^t(a) \in (0,1]$: → time and action dependent!

$$Q^{t+1}(a) = Q^t(a) + \alpha^t(a) \cdot [\mathrm{R_t} - \mathrm{Q^t(a)}]$$

If reward functions were stationary:
For which step size functions $\alpha^t(a)$ can we guarantee $Q^t(a) \longrightarrow q^*(a)$ for $t \longrightarrow \infty$?

Weighted Averages: Convergence for stationary reward distributions

- Condition for convergence of ϵ-Greedy with **stationary** rewards and $\epsilon > 0$:
  (Or any other strategy that pulls all arms infinitely many times as $T \to \infty$)
  We can guarantee $Q^t(a) \longrightarrow q^*(a)$ for $t \longrightarrow \infty$ if for all $a \in [k]$
  1. $\sum_{t=1}^{\infty} \alpha_t(\mathrm{a}) = \infty$ and
  2. $\sum_{t=1}^{\infty} (\alpha_t(\mathrm{a}))^2 < \infty$

- Constant step size $\alpha \in (0,1]$: Condition 2. is not satisfied!
  → No convergence to true action-values.
  → But: still good for non-stationary rewards.

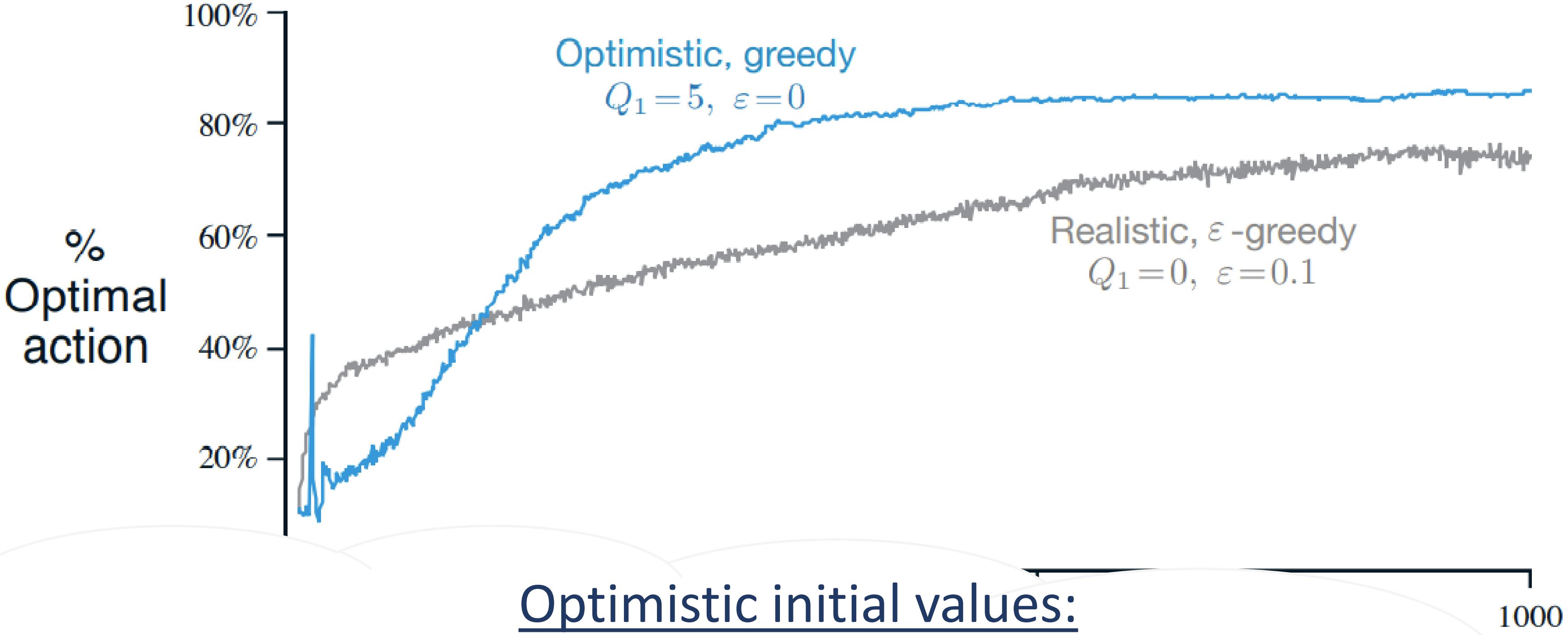
- Non-Constant step size $\alpha^t(a) \in (0,1]$:
  Example: $\alpha^t(a) = 1/N^{t+1}(a)$ [sample-average method]
  → Guaranteed convergence to true action-values by condition 1. and 2.

Optimistic Initial Values

- Let $\epsilon \in [0,1]$.
- Initialise $\mathrm{Q}^0(a) = const.$ for all $a \in [k]$.
- Initialise $N^0(a) = 0$ for all $a \in [k]$ the number of times each arm has been pulled.
- For $t = 0 \ldots T -$
  - With probabilit
  - With probabilit … andom from $\{1, \ldots, k\}$
  - Receive reward R
  - $N^{t+1}(A_t) \mathrel{+}= 1$
  - $Q^{t+1}(a) = \begin{cases} Q^t(\\ Q^t \end{cases}$

- Initial values create "bias"
  → for sample average: disappears after all arms have been selected once
  → for const. α bias persists but decreases over time
- Can incorporate prior knowledge on the arms
- Can be used to incentivize exploration in the beginning → setting *optimistic* values

Optimistic Initial Values (see Chapter 2.6 in [Link])



<u>Optimistic initial values:</u>

- exploration is only encouraged in a few initial rounds
- only useful in stationary distribution
- if reward distributions change, new exploration might be necessary

Upper-Confidence-Bound Action Selection

- Idea: Select actions that are “uncertain”, but “promising”.
- Principle: Optimism in the face of uncertainty!
  → Find upper confidence bounds on the value estimates and choose the arm with the best bound:

$$A_t = \arg\max_a Q^t(a) + c\sqrt{\frac{\ln(t)}{N_t(a)}}.$$

  → Upper confidence bounds get tighter provided more data.
  → Intuitively, we will not select a suboptimal arm too often.
- Can be implemented such that the regret is:
  $\mathrm{regret}_{\mathrm{T}}(UCB) \in \mathrm{O}(\sqrt{k \cdot T \cdot \log(T)} + \sum_{a\in[k]} q^{max} - q^*(a))$. → close to optimal!

… more on that next time!

Let’s continue the Quiz…

… go to Mentimeter!