Applied ML/RL
for Ads/Recommendation
(with Geo-spatial use cases)

Kevin Noel-Kubo

(Machine Learning Lead
Mapbox Japan)

Presentation Plan

Presentation Plan

Intro/Context

1) Online Advertising context

2) Overview of applied ML/DS in Online advertising

A) Overview of Tech challenges

B) Data Science vs Machine Learning use cases

C) Data stack: Overview and some Geo-Spatial cases.

3) Applied RL in online Ads

A) Overview of RL (vs ML)

B) RL for Ads

C) Real World examples

D) Evaluation methods

E) Overview of Tech Stack

4) Takeaways

Intro/Context

Introduction/Context

Context/Background

More than 13 years of cross-industry experience (more than 10 years in Japan).

- Worked in financial industry : Quantitative Fields (BNP Paribas, Bank of America, ING)
  - ✓ Expertise in stochastic processes / probability modelling (RL)

- Switch to ML/Big Data in Japan: work on DS/ML over past last 7 years.
  (Principal ML Engineer role, in Tokyo, Japan, Rakuten group)
  - Ads/Rec with Catalog of 500 million items, 50 mio users (Largest E-Commerce in Japan).
  - Big Data ML
  - Real time focus

  A couple of talks on Applied Machine Learning (New York (2019) , …)

- Currently, leading ML Ads Recs for Geo-Ads solution at Mapbox Japan (from end 2022).

Context: Mapbox Japan

Mapbox is a global Geo-Spatial/ Map analytics provider to corporates.

- ✓ 700 million+ MAU using Mapbox maps (ie worldwide geo-map).
- ✓ 45,000+ Mobile apps using Mapbox maps. (*>3 million registered developers*).
- ✓ Softbank is major investor (No 2 Telecom company in Japan).

Mapbox Japan

Develop custom Geo-Tech solutions for corporates (Automotive, Logistics, Retail, …)

Develop custom **Geo-Ad solution for corporates** (Yahoo Japan, …).

➢ Focus on **Real Time and large scale** geo-services to other corporates.

Context: Mapbox Japan

More details (press ...):

Japan : mapbox.jp/news

US/World: mapbox.com/about/company/#press

QA : kevin.noel@mapbox.com

1) Online advertising context

1) Online advertising context

- Ad spending in the Digital Advertising market is projected to reach **679 bn USD in 2023** (Statistica source).

- In Asia/Pacific, estimates are

  236 bn USD in 2023 (with China 173 bn) mostly through mobile (*Statistica source.*)

- Market is comprised of very large Tech actors: Google, Meta, Amazon... (US)

  but, also a multiple of big/medium/small actors across countries, by sub-industries:

  Yahoo Japan (Japan), Baidu (China),

- All are investing heavily in online Ad Technologies are different level: **Data stack, ML stack (ie Pytorch).**
- Ad Tech was major drivers of **Big Data trends** and now of **Machine Learning Tech** development.

1) Online advertising context: Japan

- In Japan, numbers Online Ads spending estimates are 25 bn (2023 estimate, Statistica).
- Largest online Ad publishers are
  - ✓ Yahoo Japan (Softbank Group, i.e. user traffic on par with Google)
  - ✓ Google
  - ✓ Line (largest social media app, Softbank Group)
  - ✓ Rakuten (E-Commerce, Multi-services)
  - ✓ Amazon Ads (E-Commerce)

    …
- Japan has a very large retail consumer base and active online ads.

1) Online advertising context: Geo-targeting

- From Business viewpoint:
  - ✓ A very large player (G Map) and a wide range of specialists by:

    Country, E-commerce activities, Mobile app (social media, navigation…)
  - ✓ Online to Offline: Bring online traffic (mobile, web) towards physical location.
  - ✓ Offline to Online: Inversely bring location search into online (i.e. E-Commerce).

- From Tech perspective:
  - ✓ Need to combine Online Ads Tech stack with Geo-Spatial data/Tech (i.e. **separate domain fields**).
  - ✓ Additional constraints to manage/connect frameworks in near Real Time.

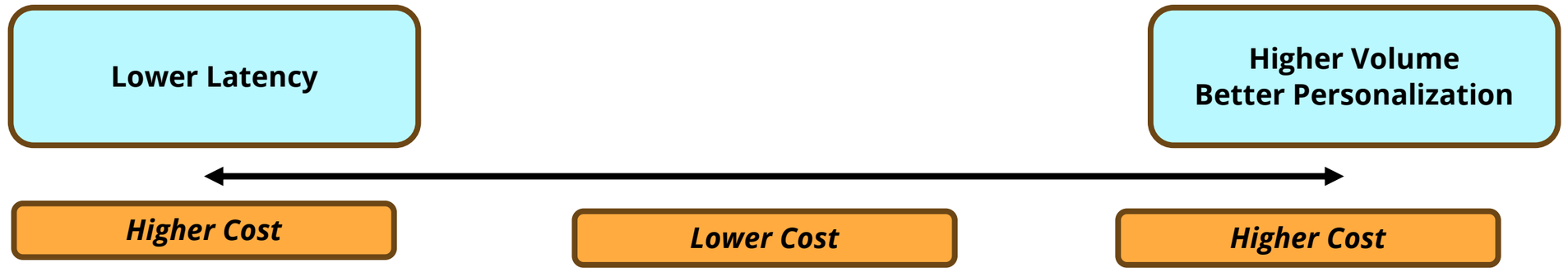
2) Overview of applied ML/DS in Online advertising

2) Overview of applied ML/DS in Online advertising

A) Overview of Tech challenges

2-A) Overview of Tech challenges

- Key Tech Pillars for Online Ads:
  - **Scalability:** Need to handle potentially a large user base: from 100,000's to billions...
    a large inventory of items: from 100's to millions
  - **Latency:** Response time in milliseconds to display
  - All keeping cost low (CPM)



2-A) Tech and ML challenges

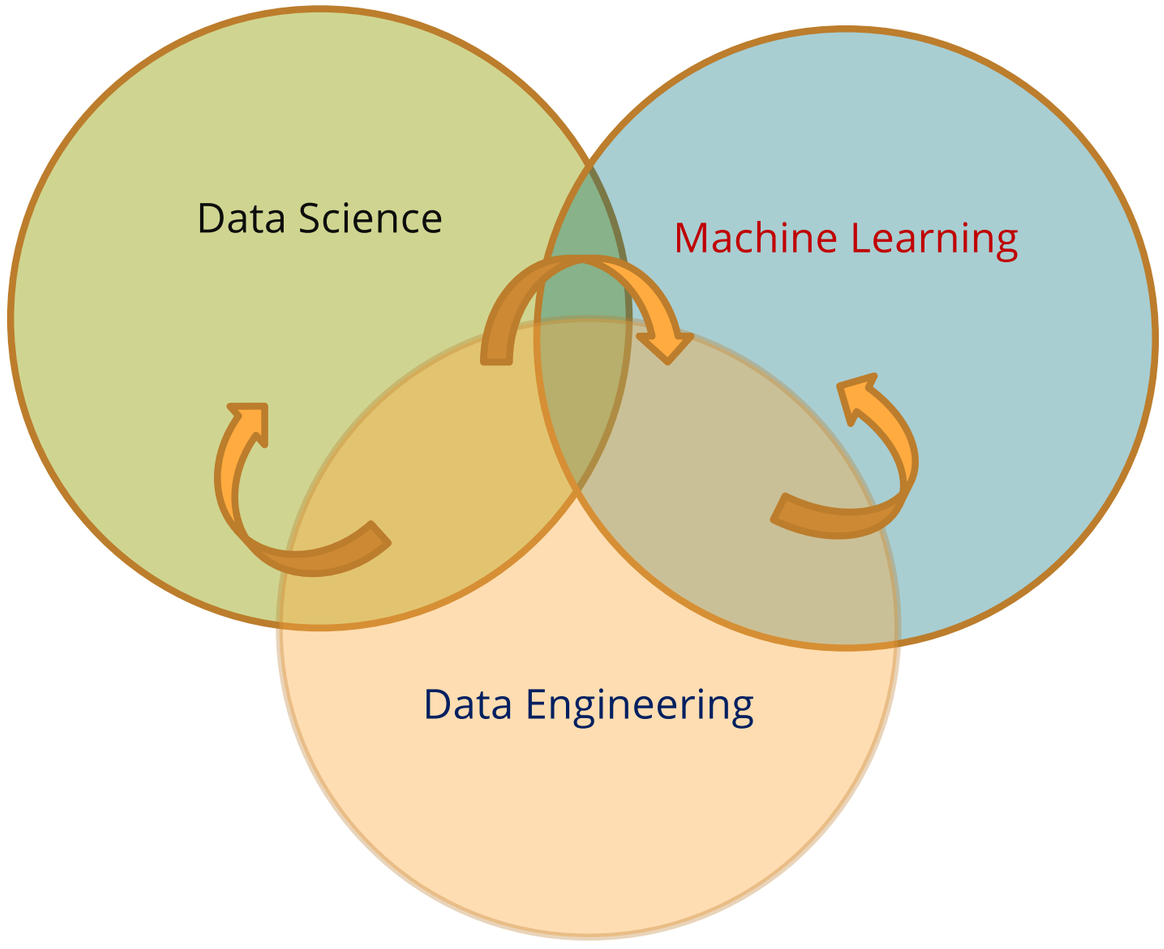
- Tech challenges are also translated into Machine Learning challenges:
  - Latency and Scalability
  - Significant Complexity addons for accurate Ads Personalization adds (ex: GPU for million of users…)

- Using ML, adds 10x challenges:
  - Higher latency
  - Higher Compute needs (i.e. ML compute is CPU/RAM bounded)
  - Re-Design Data architecture to handle a variety of ML data feed.
  - Constraints on model type, size, complexity.
  - Addons monitoring on model quality/performances…

2) Overview of applied ML/DS in Online advertising

B) Data Science and ML use cases

2-B) Data Science/ML use cases



- **Evaluation**
- **Decision Science**

Automated Intelligence (i.e. Prediction/ Recommend.)

Data Flow Pipelines

2-B) Overview of ML/DS in Online advertising

Field	Responsibility example
Data Science	• Evaluation and estimation methods (A/B test, Offline tests, ...) • Statistical methods • Domain knowledge/ Hypothesis from data → Decision Science : human based methodology
Data Engineering	• Organize data pipelines: storage + flow (i.e. daily to real time). • Feed with all data consumers (people or systems)
Machine Learning	• Focus is automation • Productionization (i.e. connect with all sytems: data, compute infra) • Science: ensure validity/correctness of ML pipelines

2-B) Data Science use cases (Ads/Rec field)

➢ Pre-Experiments/Campaign :

- Feature engineering to feed various Machine Learning:
    - Recommenders, User segmentations…
- Analysis of data for user segmentation
- Methodology for evaluation: A/B tests, KPI…

➢ Post-Experiments/ Campaign:

- Analysis of A/B tests (randomized experiments): Estimates + Causal identifications.
- Setup new hypothesis and new experiments:
    - New input features for models…
    - New causal factors for KPI…

2-B) Machine Learning use cases

- **Ads Recommender System**: not one model, a myriad of inter-connected systems.

- **Processors/ models per data type**
  - Text data → Sparse/Dense tensors
  - Image data → Sparse/Dense tensors
  - Time Series → Sparse/Dense tensors
  - Tabular data → Sparse/Dense tensors

For one data type, there are 10's different processor types and need to be trained (i.e. ML type processor).

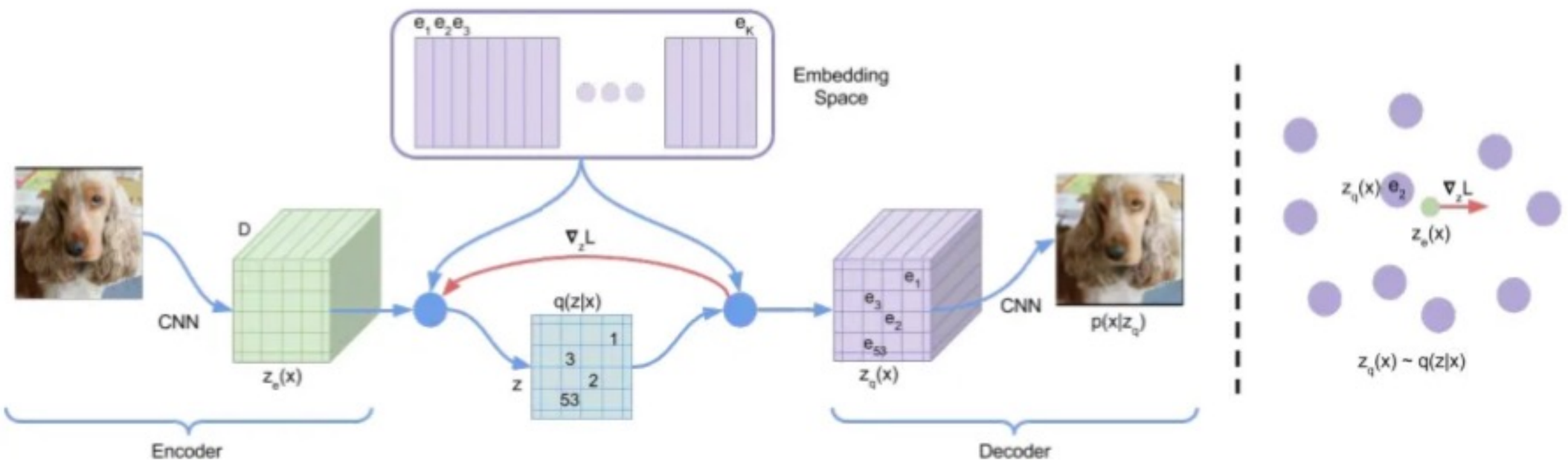
- **Inference approaches :**

  Sequential based
  Session based
  Context based
  Graph based
  Reinforcement Learning based (focus of today's talk)
  …

2-B) Machine Learning use cases

➢ **Example of one data processor** *(for image vectorization)*

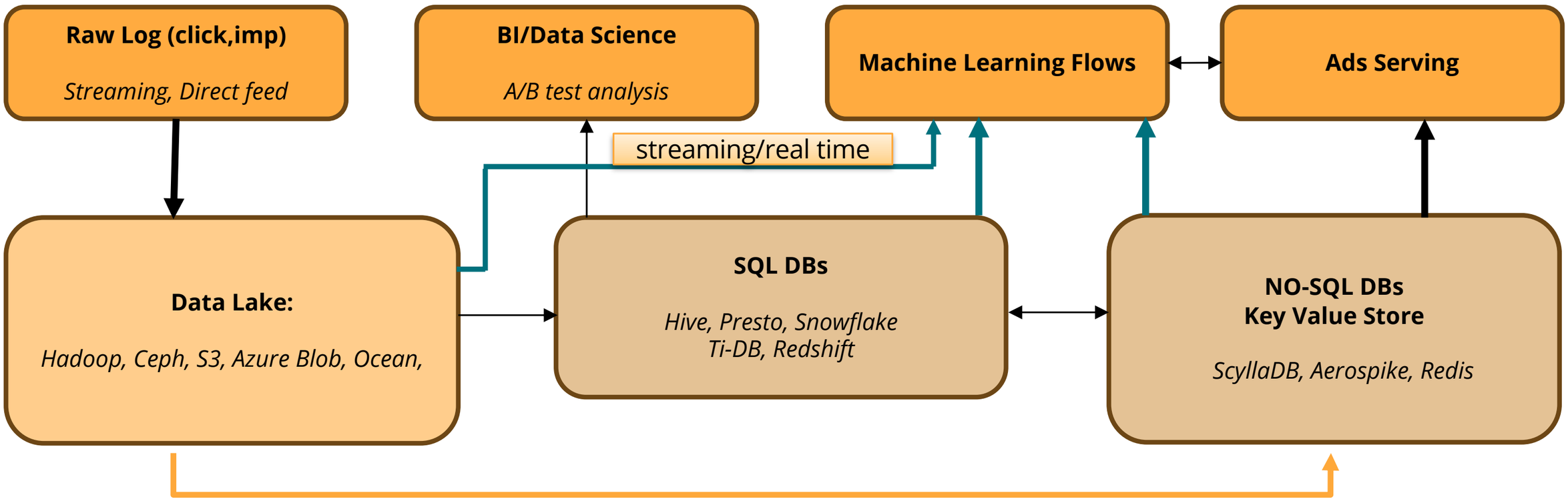
VQ-VAE neural net to semantically vectorize images data
(used for fashion image fashion data embedding)



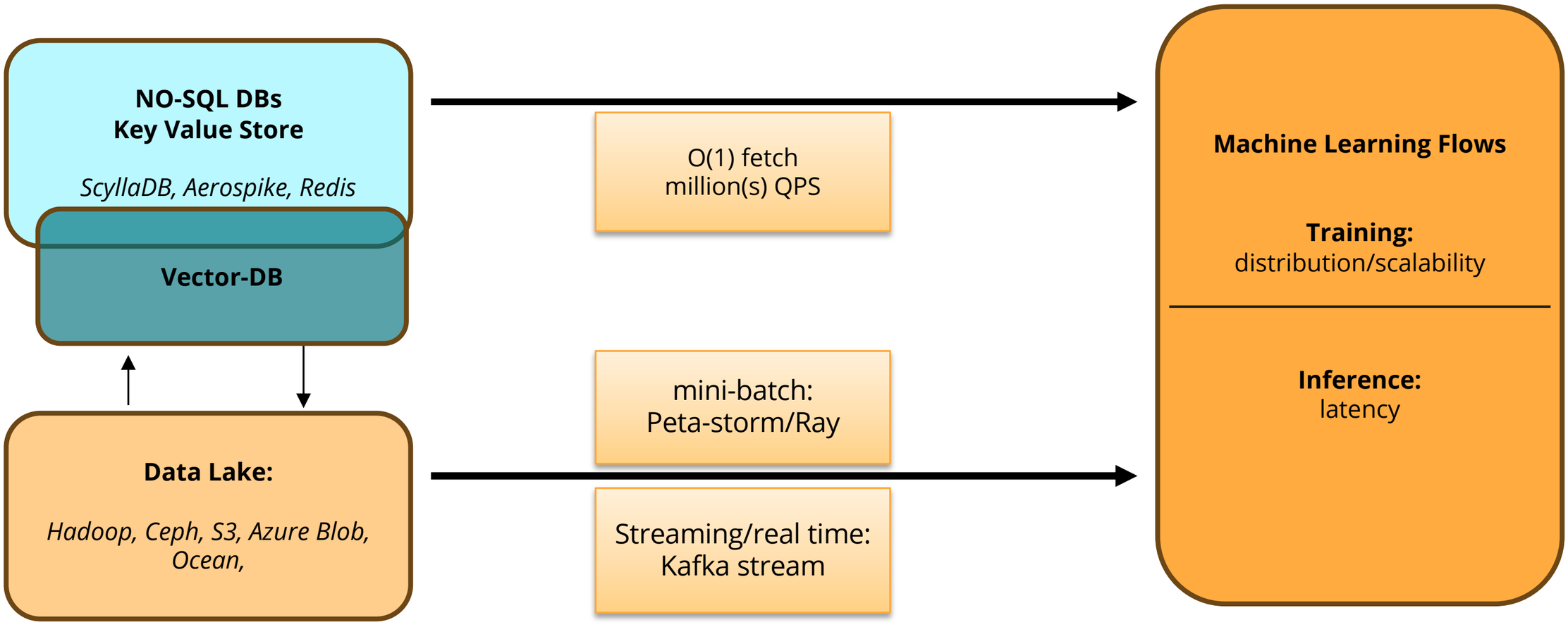
2) Overview of applied ML/DS in Online advertising

C) Data Stack: overview and Geo-spatial case

2-C) Data Stack: overview



2-C) Data Stack: overview

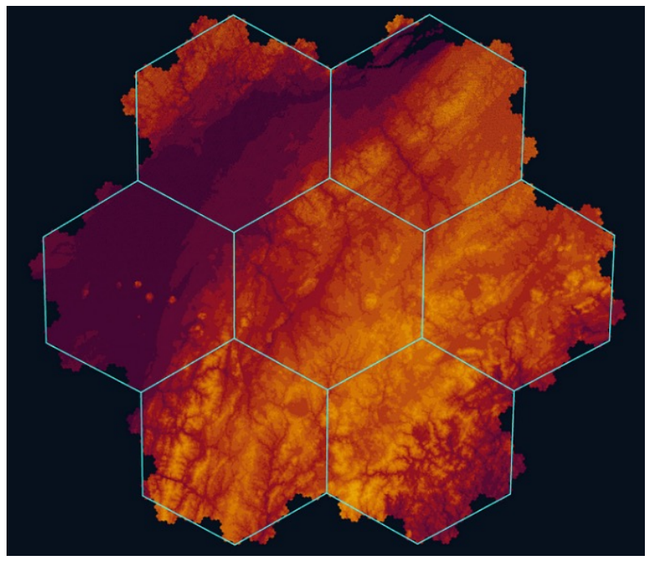


KVS and Data Lake are both scalable to peta-scale level
at lower cost

2-C) Data Stack: overview and Geo-spatial case

❖ **Concept of tile system :**

- ➢ Partition the map “uniquely” for fast identification/fetching.
- ➢ One area can be identified with different zoom levels.
- ➢ Indexation be consistent with neighborhood area (continuity).



2-C) Data Stack: overview and Geo-spatial case

Tile System	Description	Pros	Cons
Quadkey	Square Tille indentified by 64 bits integer.	1 dimenstional key Keep proximity in XY space. Hierarchical organization.	Small areas are represented by high digits index (i.e. expensive). Distortion
Uber H3 Index	Hexagonal grid	Optimal shape (i.e. Bees hexagone) Smallest number of neighbors Fast identification.	Distortions. Edge case at Poles.
Google S2 geometry	Use Spherical geometry. Consistency with earth projection.	More accurate for areas. 64 bit integer, fast indexing	Complexity and limited API

3) Applied RL in online Ads

3) Applied RL in online Ads

3-A) Overview of RL

3-A) RL overview: Learning with feedback Loop

- Most methods standard Machine Learning methods with tabular data refers:

Name	Goal	Require	Cons
Supervised Learning	Match existing Labels	True Label (i.e. costly/slow)	Cannot adapt to new data regime
Un-supervised Learning	Find ”pattern/cluster” in the data	N/A	Pattern recognition, not prediction.

- However, many ways other to learn from Data: **Reinforcement Learning** is one of them
    - Adaptative Learning: **Prediction as Action** (Intervention) on Environment and **Feedback Loop**

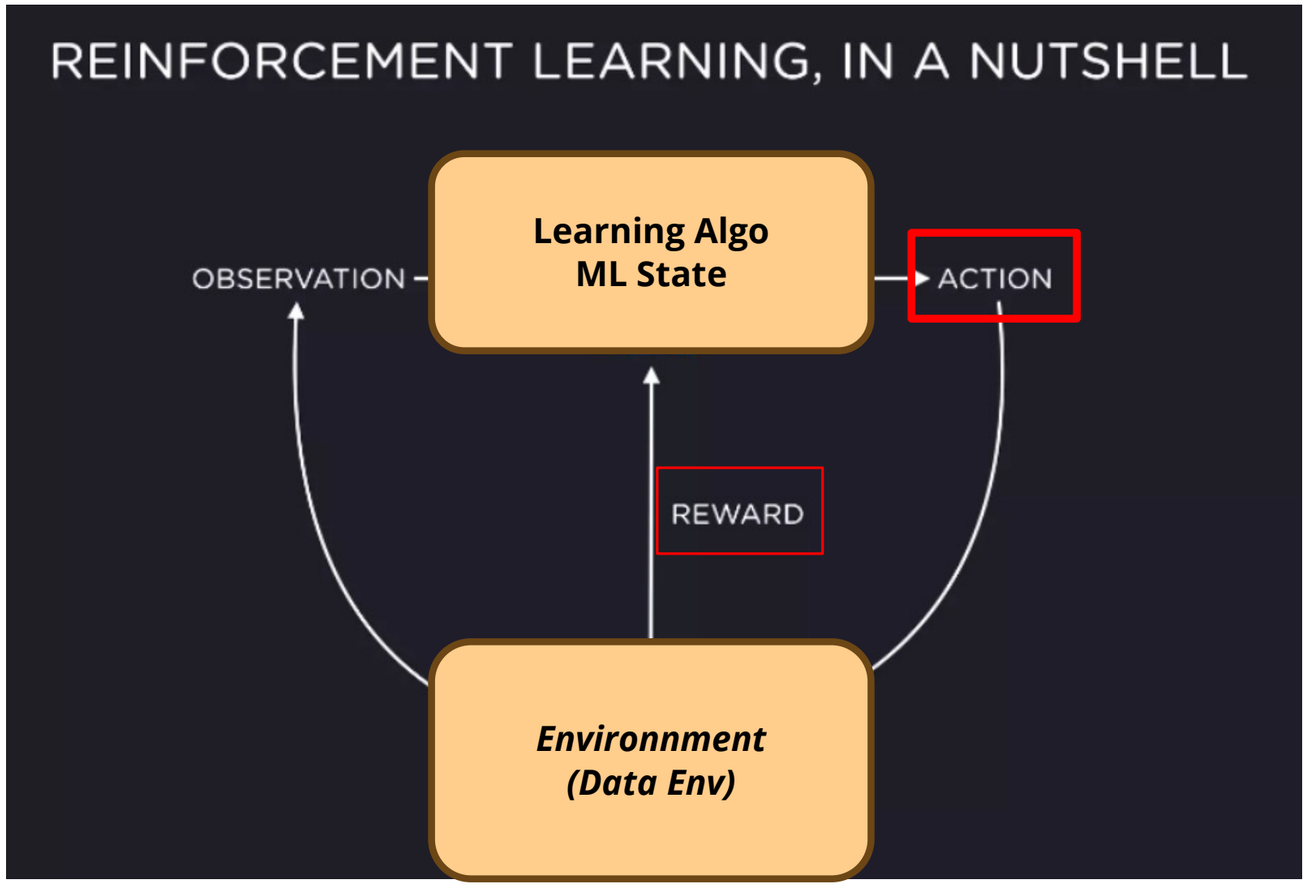
3-A) RL overview: Learning with feedback Loop

RL :

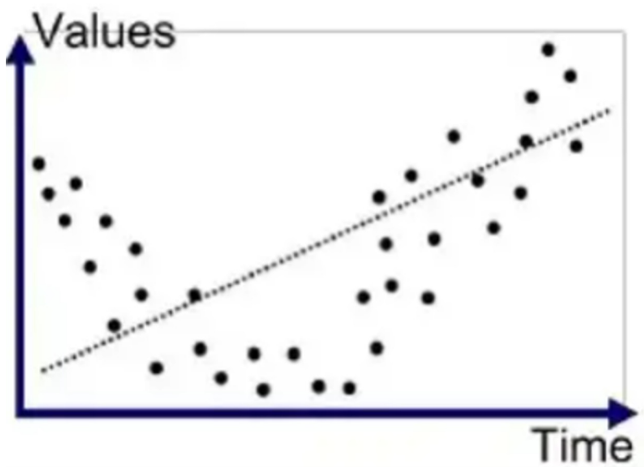
A Learning process who **dynamically interacts** with an Environment.

vs **Supervised Learning**:

An Oracle select the correct samples



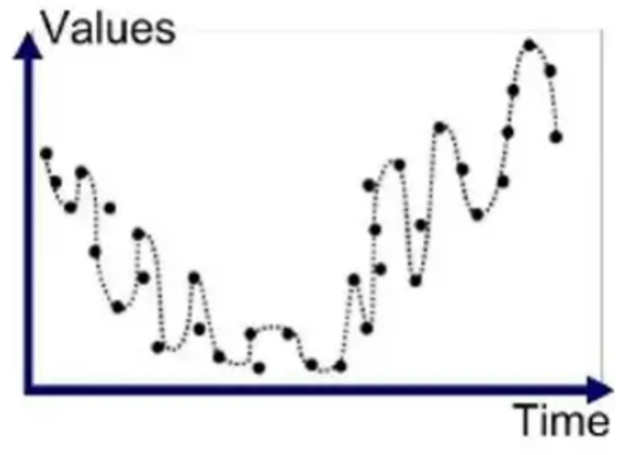
3-A) RL Overview: Benefits



Underfitted

Typical case with Real Time env.
(i.e. simpler models)

Handles better Data/Env drift (i.e. no- stationary) due to dynamic learning proces (**Action-Reward** signal)

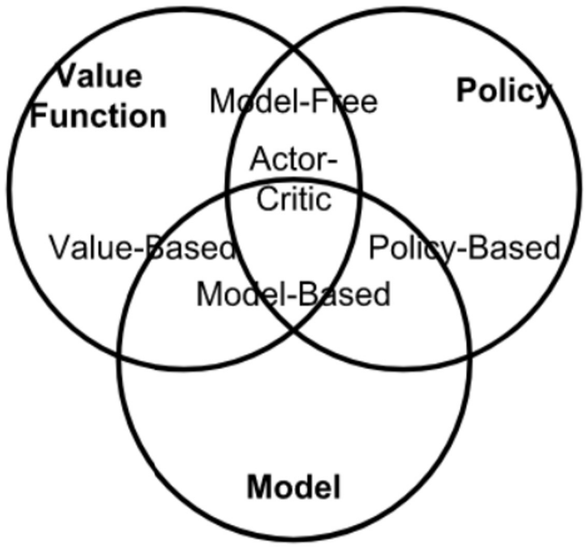


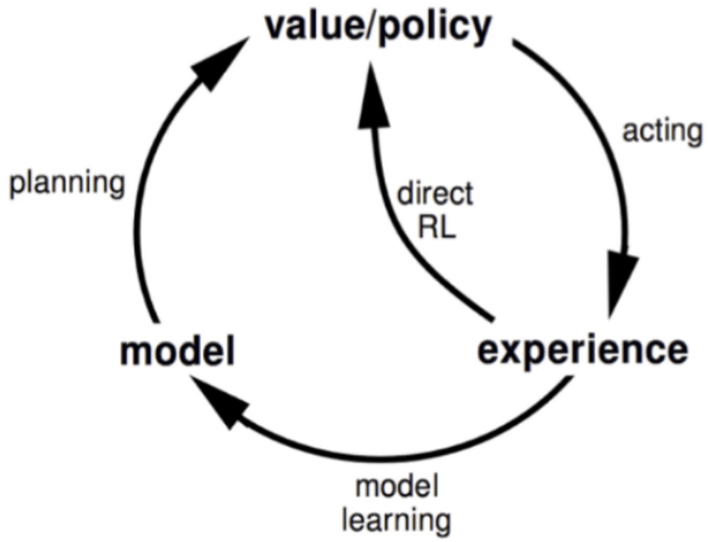
Overfitted

Typical case with
Supervised Learning

3-A) RL Overview: Include diverse academic field(s)

Many **“schools”** of RL





(ref: from D. Silver’s)

- **Model based RL:** Called also **Stochastic/Optimal Control** (i.e. in **Optimization Field**), Bellman, HJB Optimization Dynamic Programming (i.e. **DP problems**)
- **Model Free:** No explicit env modelling): Alpha Go, Zero…

3-A) RL Overview: Involves Stochastic framework

Policy

Policy, as the agent's behavior function π, tells us which action to take in state s. It is a mapping from state s to action a and can be either deterministic or stochastic:

(i.e. Decision "algo")

- Deterministic: $\pi(s) = a$.
- Stochastic: $\pi(a|s) = \mathbb{P}_\pi[A = a|S = s]$.

The reward function R predicts the next reward triggered by one action:

$$R(s, a) = \mathbb{E}[R_{t+1}|S_t = s, A_t = a] = \sum_{r \in \mathcal{R}} r \sum_{s' \in \mathcal{S}} P(s', r|s, a)$$

Total Reward $G_t = R_{t+1} + \gamma R_{t+2} + \cdots = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}$

(i.e. Discounted future reward)

3-A) RL Overview

The **state-value** of a state s is the expected return if we are in this state at time t, $S_t = s$:

$$V_\pi(s) = \mathbb{E}_\pi[G_t|S_t = s]$$

the **action-value** ("Q-value"; Q as "Quality" of a state-action pair as:

$$Q_\pi(s,a) = \mathbb{E}_\pi[G_t|S_t = s, A_t = a]$$

Goal of RL is to find a Policy (decision algo):
produces Max. expected Total Reward (i.e. for all possible future paths):

$$\pi_* = \arg\max_\pi V_\pi(s), \pi_* = \arg\max_\pi Q_\pi(s,a)$$

3-A) RL Overview

- Addons assumption (to make it solvable):
  - Markov property: MDP: depends on on current state at time t only
    Bellman Decomposition: Best Reward = Current + Discounted Future Rewards
- Many approaches to find the Optimal Policy (i.e. literature):
  - Q-Learning: Learn/Approximate Q(state,action) by Simulation type: MonteCarlo, Temp, Difference. (can use a Neural Net to store the approximation).
  - Policy Gradient: Learn the Optimal Policy directly $\mathbb{P}_\pi[A = a|S = s]$, using Policy Gradient (i.e. Reinforce)

    TotalReward = Sum[*P(Action |State)* * *Q(State, Action)* ,...]
  - A lot of others: Actor-Critic, A3C,....

3) Applied RL in online Ads

3-B) RL in Ads

3-B) RL in Ads/Rec: Stochastic Bandit

Cons of RL:

At Prediction time, Simulate all future possible paths → High latency, High compute cost

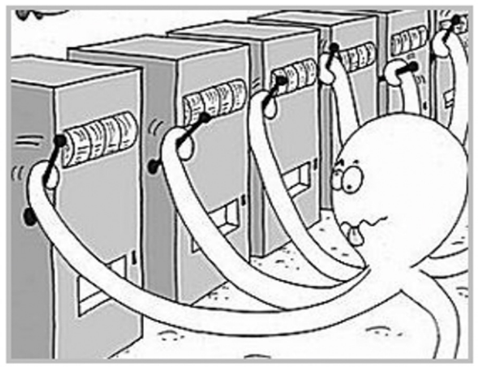
- **Simplify assumption of RL**: Stochastic Bandit (i.e. Multi Armed Bandit)
    - Remove the agent State: Focus only (Action prediction → Feedback reward).

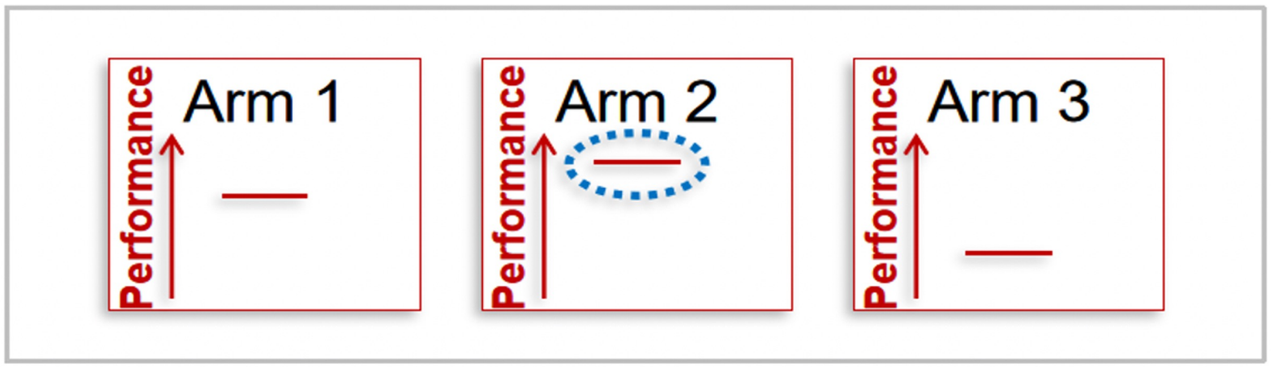
Historically, Bandit Theory has been developed outside of RL "School" (i.e. Stochastic Control…)

- Still a "standalone" sub-field, with **vocabulary different** from RL (!)
- Bandit and RL have often different domain application (!): Online Rec vs Robotics/Game/Optimization.
- Bandit is "mostly" focused on **(Actions → Instant Rewards).**

3-B) RL in Ads/Rec: Stochastic Bandit

- **Bandit Framework**: Simplify RL assumptions
    - RL with Agent State is Constant.
    - Only focus on (**Action, Reward**) pairs.





Action (in RL) = **Arm** (in bandit language)

Action is called Arm (in bandit language).

3-B) RL in Ads/Rec: Stochastic Bandit assumptions

- Agent has constant state (state=Cst).
- Reward until current time is used (i.e. not future reward): Simplify credit assignment (reward to which action...).
- Action Space is Fixed (Discrete): K Actions with unknown reward (i.e. often a probability prior distribution is assigned).

<u>RL setup is simplified</u>

Policy : $\pi(a) = P_\pi [A = a]$

Action Value Q: $Q_\pi(a_i) = \mathbb{E}_\pi[R_t | A_t = a_i] = \theta_i$

State Value V : $V_\pi = \sum_{a \in \mathcal{A}} Q_\pi(a)\pi(a)$

Goal is to find a policy which maximizes Total Reward.

3-B) RL in Ads/Rec: Bandit Regret

- In Bandit assumptions, one can prove existence of optimal policy Q*.
    - Regret: difference of reward between **Optimal policy Q*** and current **policy Q.**

Total Regret $\mathcal{R}_T = \mathbb{E}\left[\sum_{t=1}^{T}\left(Q^*(a_t) - Q(a_t)\right)\right] = \mathbb{E}\left[\sum_{t=1}^{T}\left(r(t, a_i^*) - r(t, a_i)\right)\right]$

    - Max. Total Reward equivalent to **Minimizing Total Regret**
    - **Measures the cost of wrong Exploration and wrong Exploitation over time.**

- In Bandit setup:
    - Finding good Policy focuses on finding policy with minimal (asymptotic) Regret over time horizon T

3-B) RL in Ads/Rec: Bandit Regret

Example of policy with their Regret Bound

Policy name	Reward type	Asymptotic Regret Upper Bound
Epsilon Greedy (eps=cst)	Bernoulli (1/0)	Regret < O(T) : linear, can be sub-linear
TSsampling	Gaussian reward	Regret < O(Sqrt[N_arm *T *Log(T)])
UCB	Bernoulli reward	Regret < O(N_arm * log(T))

Optimal Lower Bound is O(logT)

Only Explore Policy (i.e. switching action): O(T) (linear)

Only Exploit Policy: O(T) (linear)

3-B) RL in Ads/Rec: Contextual Bandit

- In Real world: Reward is dependent on Action type + some “State” (Context).

- Add “Pseudo State” to the agent through “Conditionalizing” the reward with some Context
    - Reward = E[Rt | **Context(t)**]
    - Context(t) = Random Vector of co-variates (ex**: time, location, embedding XXX**, ...)

- More realistic modeling of reward.

- Contextual Bandit are one the most popular approach in **Real World Ads/Recommender.**

3-B) RL in Ads/Rec: Contextual Bandit example

Thomson Contextual Bandit with linear payoff (2012, Agrawal).

Bayesian setting with **Prior/Posterior sampling** of the contextual expected reward:

$$\mathbb{E}\left[r_i(t)|X_i(t)\right] = X_i(t)^T.W$$

Xi(t): Context Vector for arm i at step t, collected from data (i.e. covariates: time, location, item, embedding...).

W: Weight Random vectors, following Gaussian Prior distribution: $Normal(\mu, \sigma)$

At each time step:

Update the posterior distribution with new feedback data : $Normal(\mu_2, \sigma_2)$

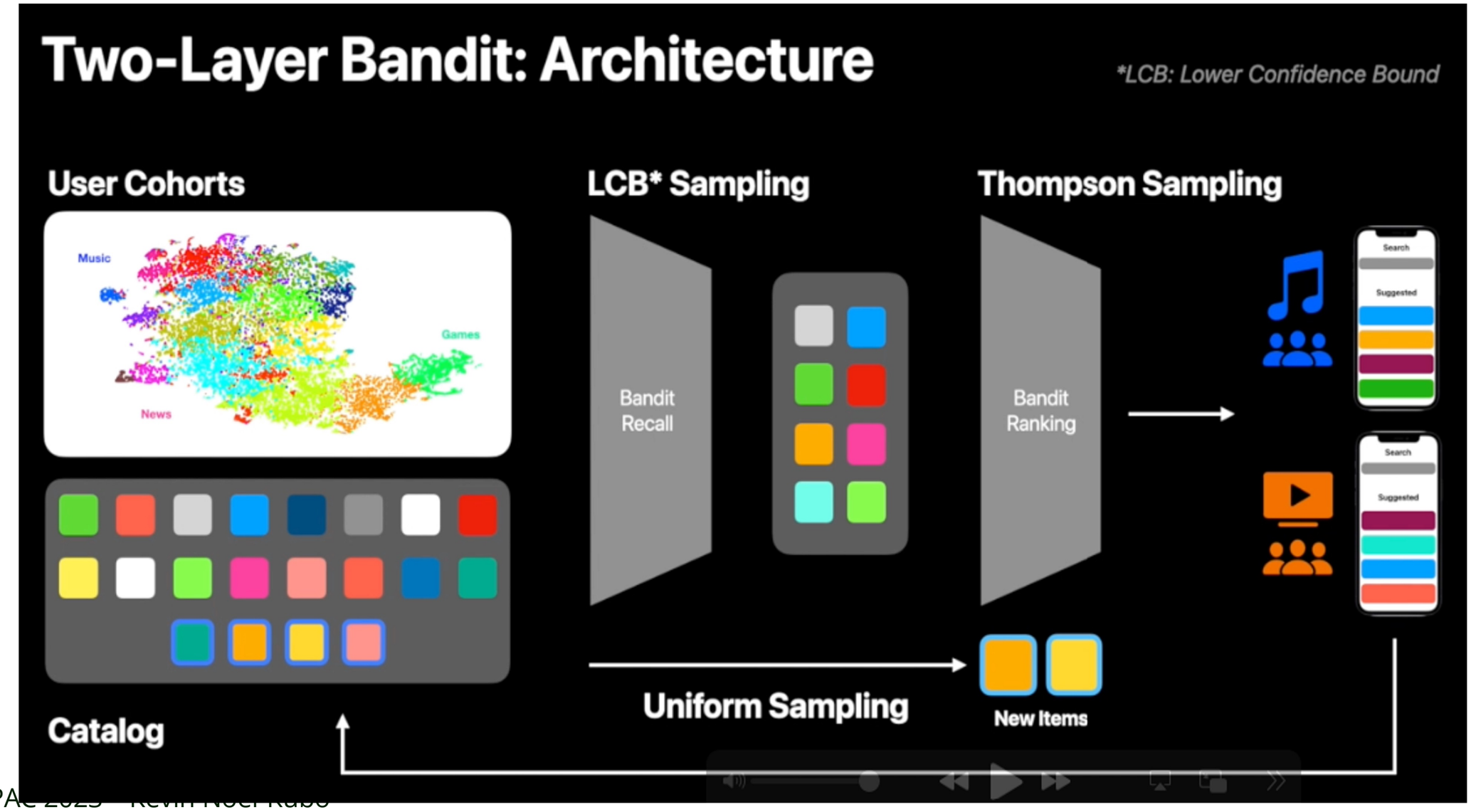
Sample vector W from posterior: $Normal(\mu_2, \sigma_2)$

Select arm i which maximize Expected reward: $BestArm_t = ArgMax\left[X_i(t)^T.Ws\right]$

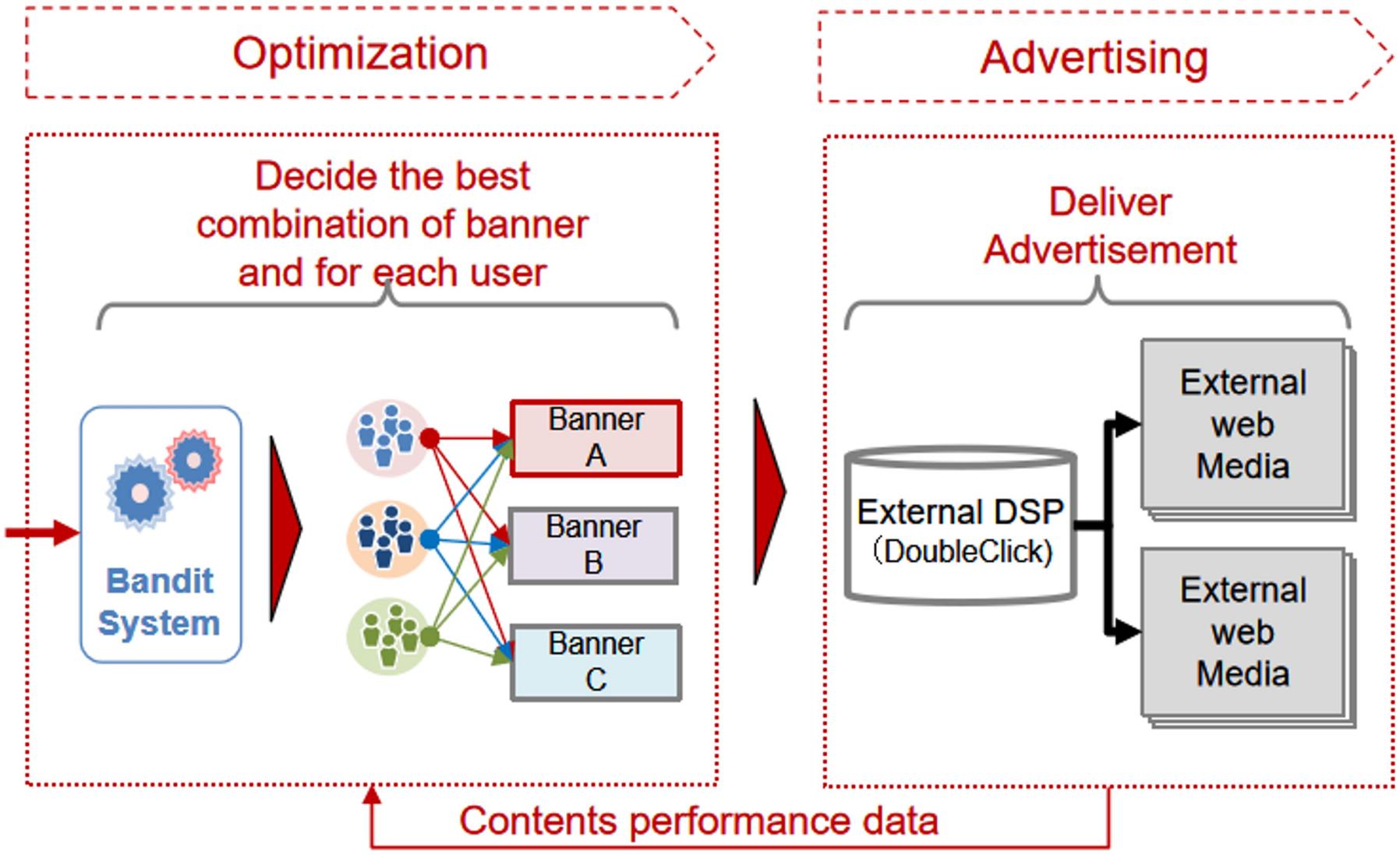
3) Applied RL in online Ads

3-C) Real world example

3-C) Example: Double Bandit for App Store app recommendation

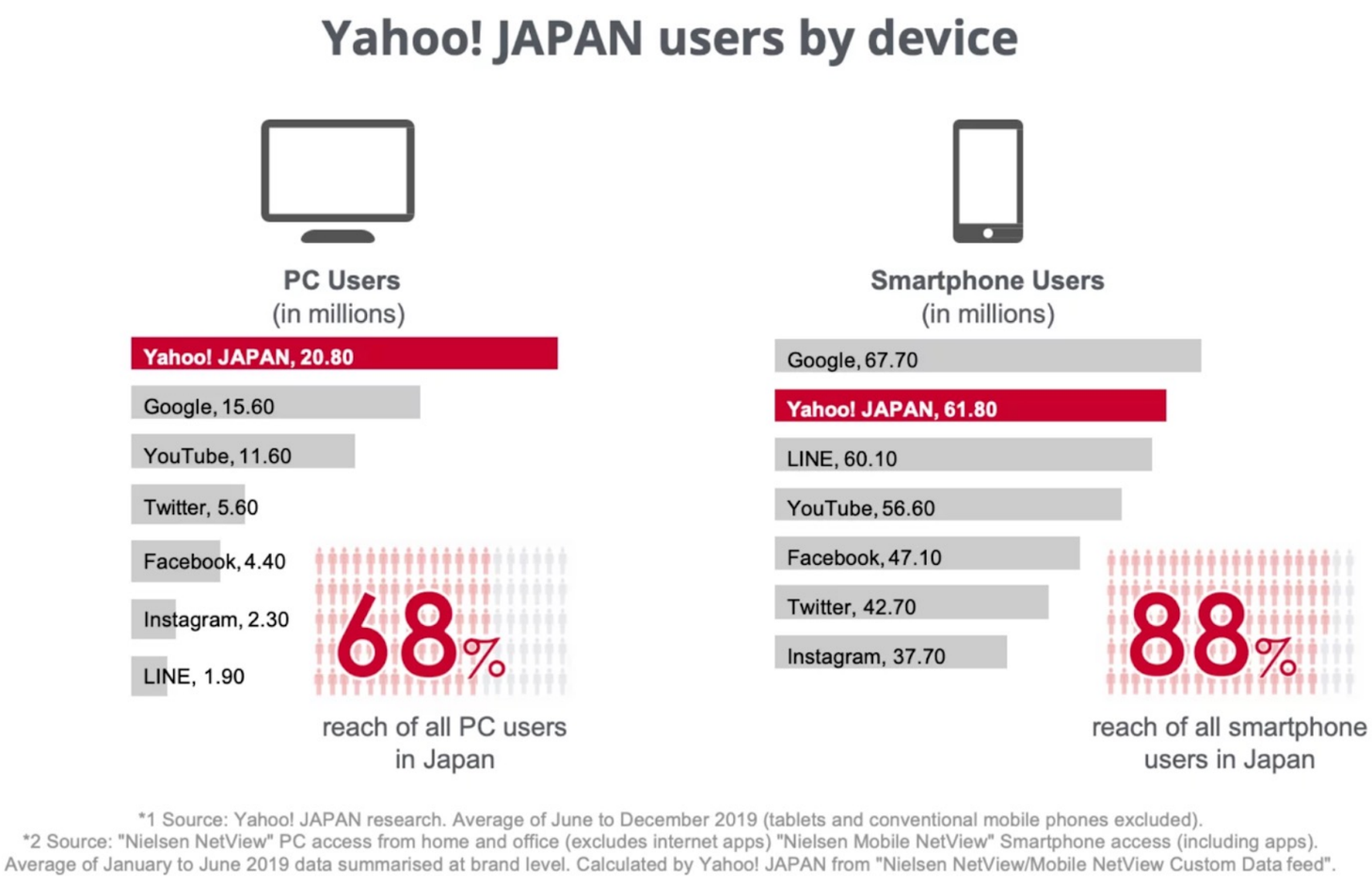


3-C) Example: Personalization



3-C) Example: Geo using Bandit-type

3-C) Context: Yahoo Japan is top 1-2 largest web traffic in Japan (i.e. equal with Google)



Yahoo Japan has popular Map application using **Mapbox**

Japan Tech.

3) Applied RL in online Ads

3-D) Evaluation methods

3-D) Evaluation methods: Introduction

- **2 types of methods:**

  Offline: Metrics using past data.

  Online: Metrics using real world through A/B Test.

  Goal of offline evaluation is to "correlate/align" with actual real world (i.e. online) metrics.

  - Often mis-match between offline/online **(i.e.: nb of clicks, $$$ amount purchase).**

- **Offline methods:**

  - **Ranking Recommender metrics**: MRR, NDCG... or pointwise metrics (i.e. but, Bias of past logs).
  - **Policy Eval**: Typically "IPS –type" method by re-sampling past data to remove Bias.

3-D) Evaluation methods: IPS method

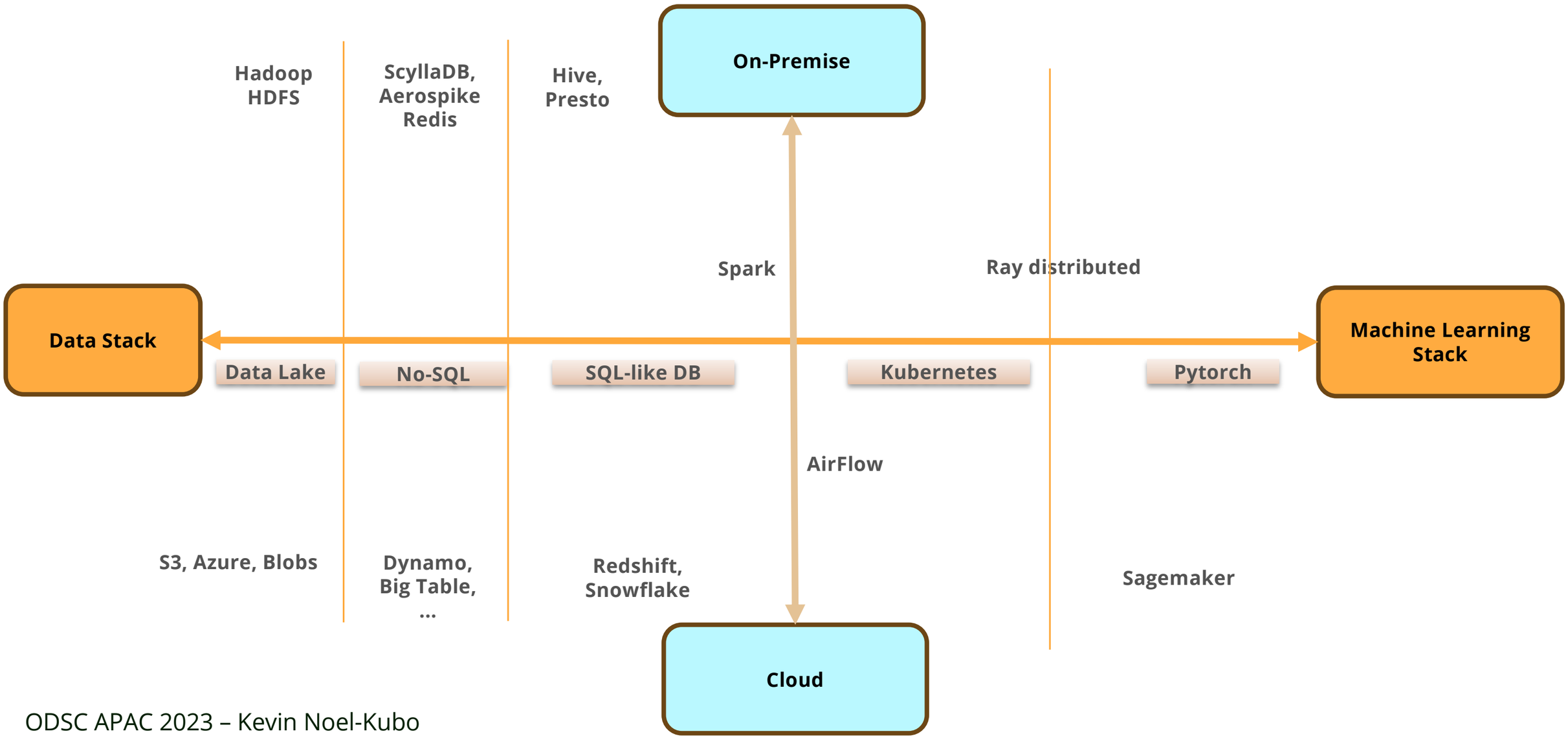
- **IPS: Inverse Propensity Score**
    - Re-Sample past (action, reward) pairs and use **weight = 1/PropensityScore** to remove Algo Bias.
- Active research to refine methods (i.e. Bias/Variance of the estimator):

	model-based	importance sampling-based	bias	variance
Direct Method (DM) [Beygelzimer&Langford,09]	✓	---	high	low
Inverse Propensity Scoring (IPS) [Precup+,00] [Strehl+,10]	---	✓	unbiased	very high
Doubly Robust (DR) [Dudík+,14]	✓	✓	**unbiased**	**lower than IPS, but still high**

3) Applied RL in online Ads

3-E) Overview of Tech stack

3-E) Overview of Tech stack: “Not one solution fits all” approach…



3-E) Overview of Tech stack: Own experience sharing

- **Currently on AWS Stack:**

  S3, Athena, EKS (managed Kubernetes), EC2, …

- **Previously On Premise:** Hadoop + Hybrid Cloud (Internal + Google CP)

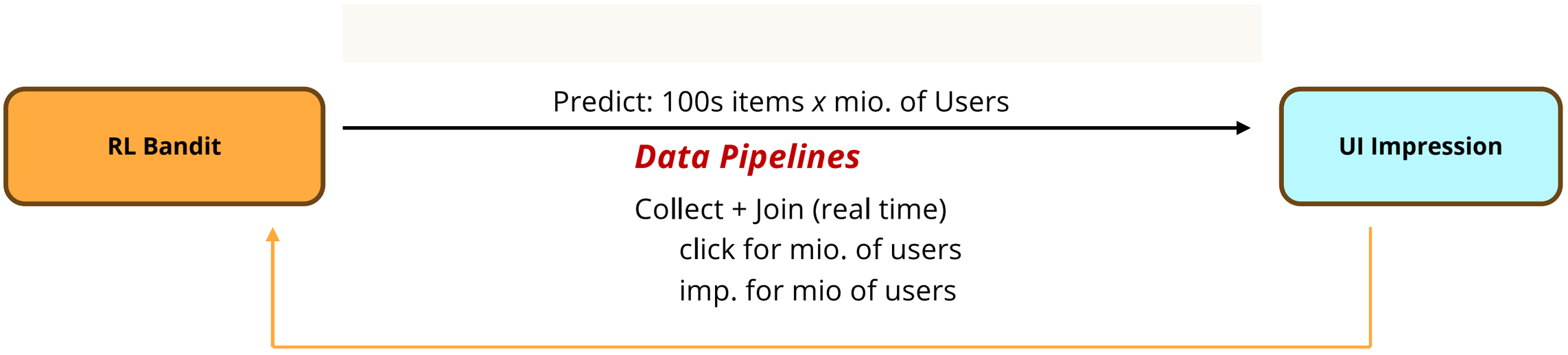
  Ceph, GLuster, Hadoop,
  Hive, HBase, Cassandra, ScyllaDB, Aerospike, Redis, …
  Kubernetes, Docker Swarm, Ceph, Gluster,…

- **Remarks:**

  - Data stack are often legacy/immutable : A lot of customization efforts required.
  - Machine Learning stacks are (often) portable:
    Container/K8s based, Cloud agnostic… but tooling are disparate

3-E) Tech Stack: What's differentiate RL from ML

- In standard ML recommendation: feedback is (often) gathered daily.
    - Merged into daily data processing (i.e. aggregation + cost is reduced).
- In RL/Bandit setup, **Feedback Loop** must be near real-time:
    - Longer delays creates **Biases in** RL Prediction + Mismatch (i.e. Delayed reward...).
    - Real time join are often "bottleneck"



3-E) Tech Stack overview

Scaling and Latency vs Metrics

3-E) Tech Stack: Scaling and latency

- **Typical requirements (Ads/Rec E-Commerce):**
    - Server 50 million active user daily, with peaks of 100,000 query/sec.
    - Select best items out of 100 items - 500 million items.
    - Server final user(s) in 50 milisec → model inference < 15ms.

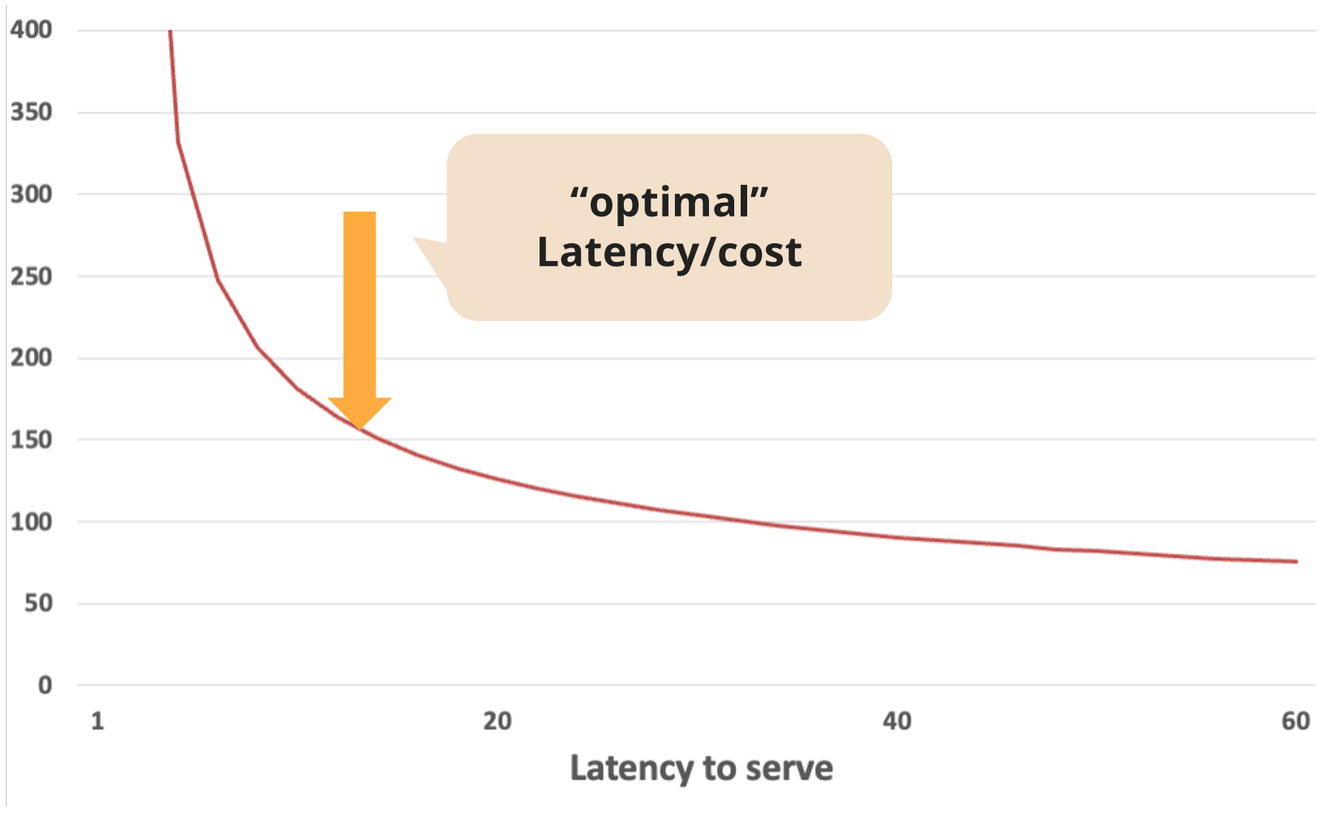
- **Real challenge: Personalization:** Per User data history, calculate relevant scores (Bandit or else).
    - Generate scores for each (user, item) pairs to re-rank them.

      per day: 50mio x 5mio.items: 250.10^{12} score/day.

      per second: 100k x 5 mio.items: 5.10^{8} score/sec.

3-E) Tech Stack: Latency cost in Ads (ML/RL)

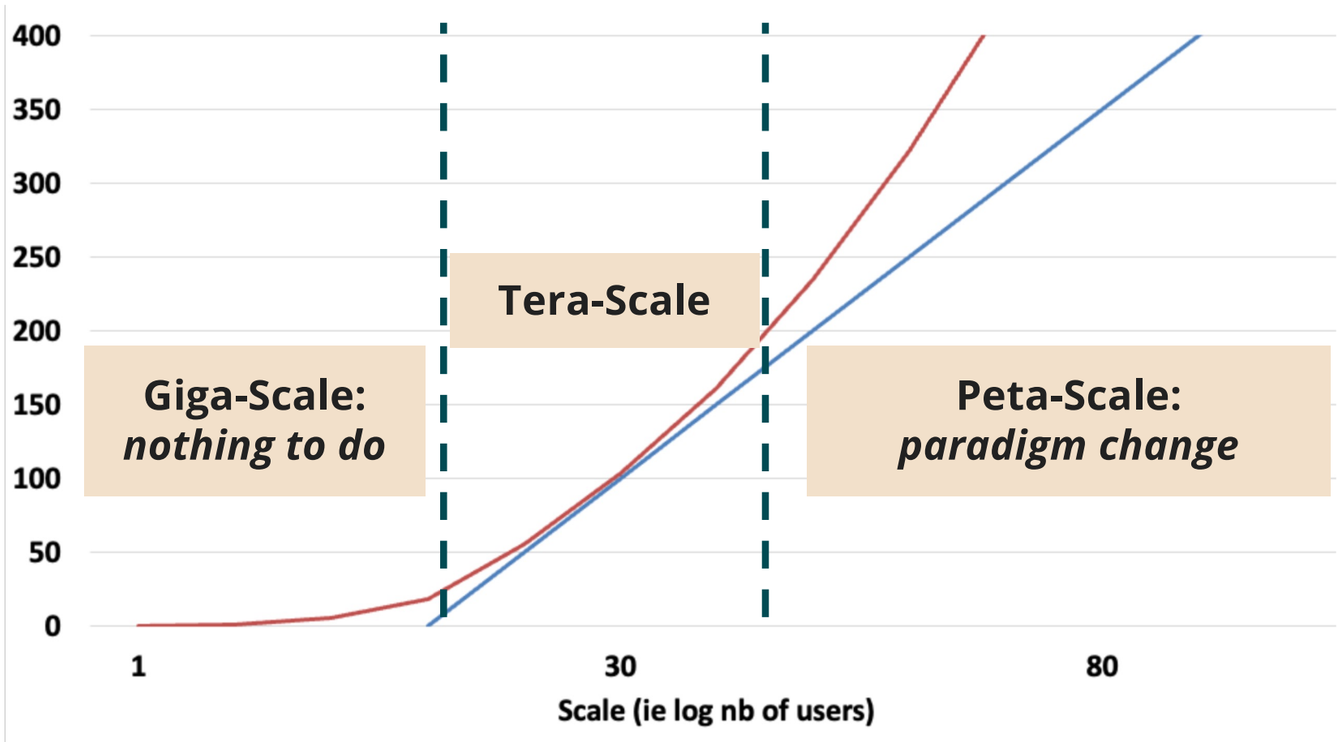
Actual Cost to serve ML



- Latency requirements constraint :
  - Model complexity
  - Data process complexity
- A lot efforts to reduce model inference latency:
  - Compile in native code.
  - Custom hardware (TPU, Inferencia,…)
- System Design to reduce overall latency:
  - Distributed architecture : Compute + Data Lake

3-E) Tech Stack: Scaling cost in Ads (ML/RL)

Actual Cost to serve ML



- System/Data infrastructure required to be change when reach some levels:
  - **GigaScale vs TeraScale vs PetaScale**
- Data infrastructure makes most of the cost...
- **Add constraints on type of model:**
  - **Not all models can scale...**
  - **Not all model is cheap enough to scale...**

3-E) Tech Stack

RL implementation: Pytorch

3-E) Tech Stack: Pytorch as core ML framework

- **Pytorch ecosystem becomes very mature:** Support of Meta/Microsoft… Big Techs.
- **Research/Dev mode:** Pytorch with python mode → Eager mode
    - Focus is flexibility of developing new models: ++ for research.
- **Real World:** Pytorch Python can be quite slow (and **costly**), Latency + Scaling…
    - Need to compile (need to re-factor the code):
        - Torch Script.
        - TVM compiler, AWS Neuron, ONNX Runtime.

3-E) Tech Stack: Pytorch as core ML framework

- **Main challenge of compiler like Torch Script:**
  - Design primarily for **NLP, Computer Vision** operators (i.e. convolution, attention…).
  - Not designed for RL perspective: **torch.distributions** cannot be compiled natively…
    - ✓ Need to rewrite torch.distributions with custom operator.
    - ✓ Ex: Sample Beta distributions with Dirichlet disitribution.

Dirichlet

CLASS torch.distributions.dirichlet.Dirichlet(*concentration*, *validate_args=None*) [SOURCE]

Bases: ExponentialFamily

3-E) ML case of Geo-Spatial data

Main challenges (i.e. example):

- Need to retrieve locations in 2 **axes** at same time.
  - **Spatial** nearest neighbor (Geo-Index + Distance).
  - **Semantic** nearest neighbor (NLP embedding).
- Geo-Spatial Index for fast pre-retrieval of (lat, lng) area
  - Quadkey, Uber H3, Google…
- NLP embedding from description + Category

4) Conclusion

4) Conclusion

Some take away:

- ✓ Online Ads a major driver of Tech innovation/investment.
- ✓ Data Science/Machine Learning in Ads/Rec: different domain fields but interconnected.
- ✓ Tech/Data stacks in Ads/Recommender: must follow strict constraints (latency/scalability).
- ✓ Integration in Map/Geo-Spatial: require specific Data stack/pipelines.
- ✓ Reinforcement Learning in Ads/Rec: brings feedback loop to adapt to user intents.

Predict Actions → Collect Feedback

- ✓ RL requires specific data pipelines, machine model data flows on top of existing Tech Stack.
- ✓ Evaluation process is fundamental in designing such systems.

QA