Linear Bandit

A Contextual-Bandit Approach to Personalized Article Recommandation, Li et al., 2010

Hippolyte GISSEROT
Maximilien WEMAERE
Yann TRÉMENBERT

Study context


A Contextual-Bandit Approach to
Personalized News Article Recommendation

Lihong Li[†], Wei Chu[†], [†]Yahoo! Labs lihong,chuwei@yahoo-inc.com
John Langford[‡] [‡]Yahoo! Labs jl@yahoo-inc.com
Robert E. Schapire[+*] [+]Dept of Computer Science Princeton University schapire@cs.princeton.edu

ABSTRACT
Personalized web services strive to adapt their services (advertisements, news articles, etc.) to individual users by making use of

service vendors acquire and maintain a large amount of content in their repository, for instance, for filtering news articles [14] or for the display of advertisements [5]. Moreover, the content of such a web-service repository changes dynamically, undergoing frequent

Movie recommendation system

	2	0	3	5	4	2
	4	3	0	0	4	5
	2	3	5	0	1	0
	5	0	4	3	5	0

Goal: predict the highest rewarded movies and recommend them to the users.

Formulation of Contextual-Bandit:

For each trial $t = 1, 2, 3, \ldots$ In trial t:

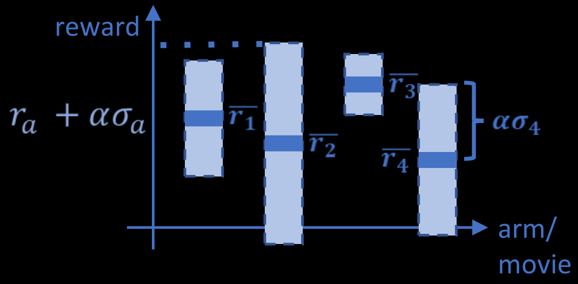
1. The algorithm observes the current user u_t and a set A_t of arms/movies together with their feature vectors i.e. **contexts** $\boldsymbol{x_{t,a}} \in \mathbb{R}^d$ for $a \in A_t$.
2. Based on observed payoffs in previous trials, we choose an arm $a_t \in A_t$, and receive **payoff** $\boldsymbol{r_{t,a_t}} \in [0, 5]$.
3. We improve our movie-selection strategy by taking into account the new observation $(x_{t,a_t}, a_t, r_{t,a_t})$.

$$Regret_i = r^* - r_{chosen\ arm}$$

$$CumRegret_n = \sum_{i=1}^{n} Regret_i$$

How to choose an arm

We calculate expected payoffs r_{t,a_t} and bounds $\alpha\sigma_{t,a_t}$ for each arm and select the arm with the highest upper bound.



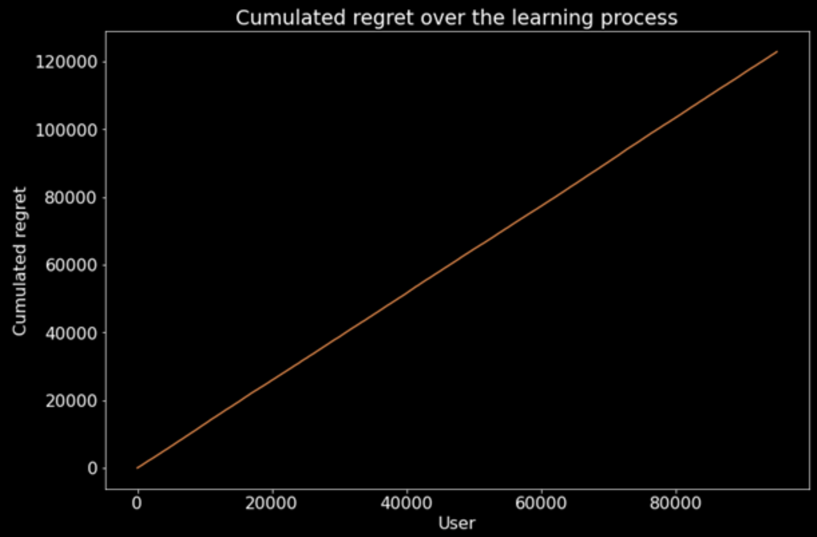
Different methods allow to calculate those expected payoffs and bounds:

→ Context-free models (random strategy, ϵ-greedy, K-bandit…)
→ Context-free linear model
→ **Contextual linear model (CLB)**
→ **Hybrid linear model (HLB)**
→ …

Reference approaches

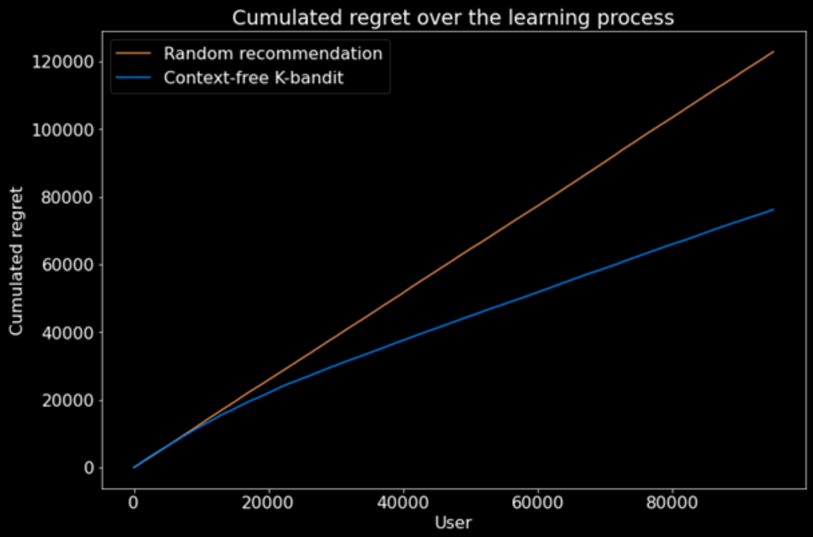
Random strategy

```
chosen_arm = random.choice(possible_arms)
```



Context-free K-bandit algorithm

$$m_i^* = \underset{m \in \{\text{movies}\}}{\operatorname{argmax}} \; \overline{r_{i,m}} + \frac{\alpha}{\sqrt{i-1}}$$



Context-free linear bandits

→ Information we have: ratings, movies' features $(x_m)_m$

→ Problem formulation:

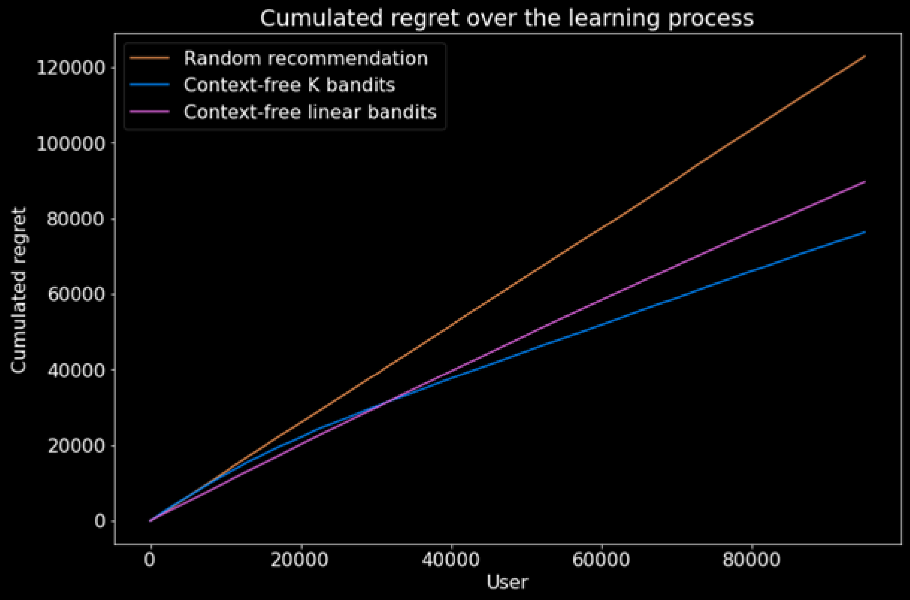
$$\mathbb{E}\left[r_{i,m}|x_m\right] = x_m^T\theta_i^*$$

→ Selection policy:

$$\underset{m\in\{movies\}}{\operatorname{argmax}} \; x_m^T\theta_i^* + \alpha\sqrt{x_m^T\left(X_i^TX_i - \lambda I\right)^{-1}x_m}$$

→ Conclusion: more efficient than K bandits in the short run but gets outperformed beyond the $3{,}000^{th}$ iteration

→ Interpretation: movies' features do not bring much information (too many?)



Contextual linear bandits (CLB)

→ Information we have: ratings, movies' features $M = (x_m)_m$

→ Information we need: users' features $U^* = (x_i)_i$

$$U^* = \underset{U}{\operatorname{argmin}} \|UM^T - R_{train}\|^2$$

→ Problem formulation:

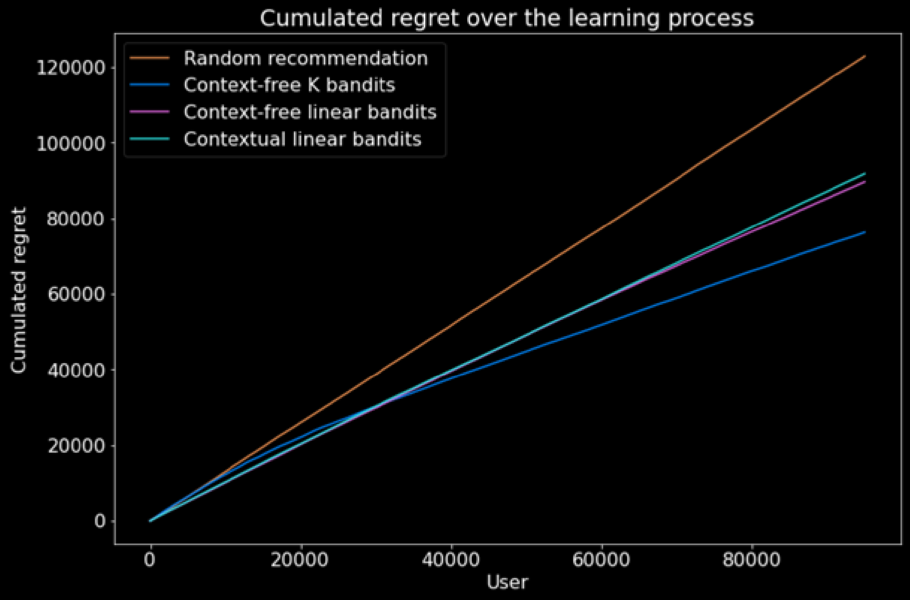
$$\mathbb{E}[r_{i,m}|x_i] = x_i^T \theta_{i,m}^*$$

→ Selection policy:

$$\underset{m \in \{movies\}}{\operatorname{argmax}} \; x_i^T \theta_{i,m}^* + \alpha \sqrt{x_i^T \left(X_{i,m}^T X_{i,m} - \lambda I\right)^{-1} x_i}$$

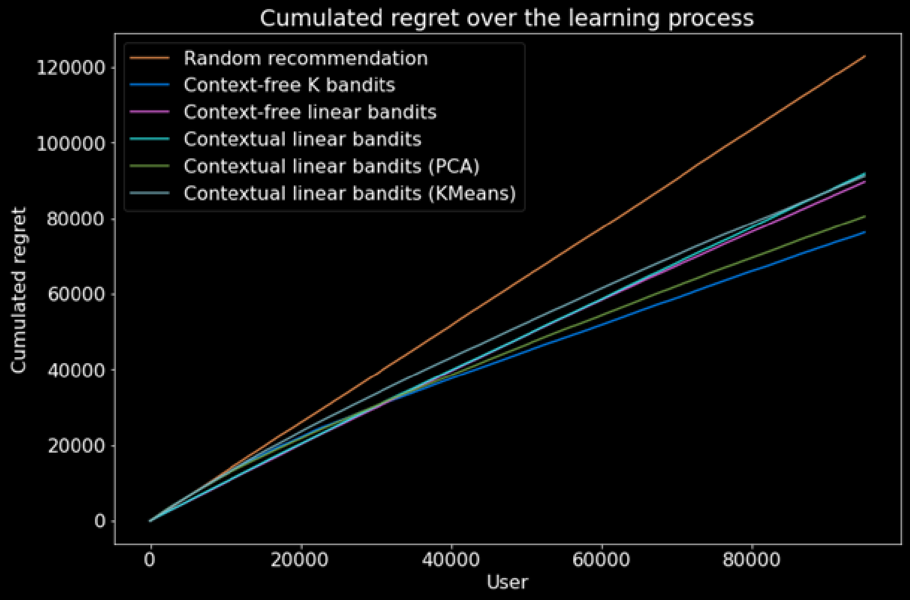
→ Conclusion: context does not bring much information

→ Interpretation: artificially built users' features, still too many features?



CLB – dimension reduction

- → Framework: exactly the same as in the previous approach, except that we want to reduce the number of users’ and movies’ features (from 21 to 5)
- → Dimension reduction methods: PCA, KMeans clustering
- → Conclusion: PCA works much better than KMeans and manages to significantly improve the performance of the algorithm, yet it remains less efficient than context-free K-bandits
- → Interpretation: artificially built users’ features, not enough data to estimate them properly



Hybrid linear bandits

→ Information we have: ratings, movies' features $M = (x_m)_m$, users' features $U = (x_i)_i$, interaction features $(z_{i,m})_{i,m} = (x_i x_m)_{i,m}$

→ Problem formulation:

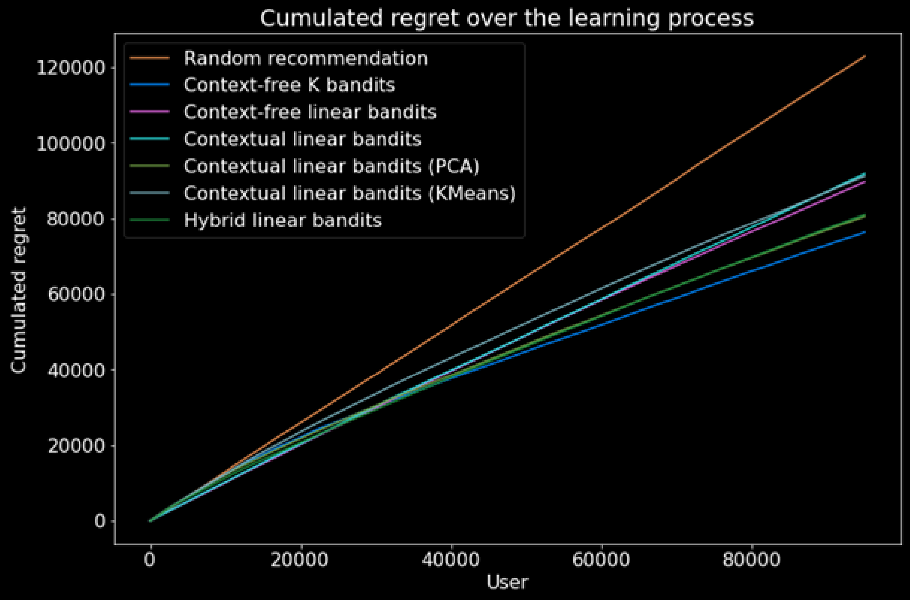
$$\mathbb{E}[r_{i,m} | x_i, z_{i,m}] = x_i^T \theta_{i,m}^* + z_{i,m}^T \beta_i^*$$

→ Selection policy:

$$\underset{m \in \{movies\}}{\operatorname{argmax}} \; x_i^T \theta_{i,m}^* + z_{i,m}^T \beta_i^* + \alpha \sigma_{i,m}$$

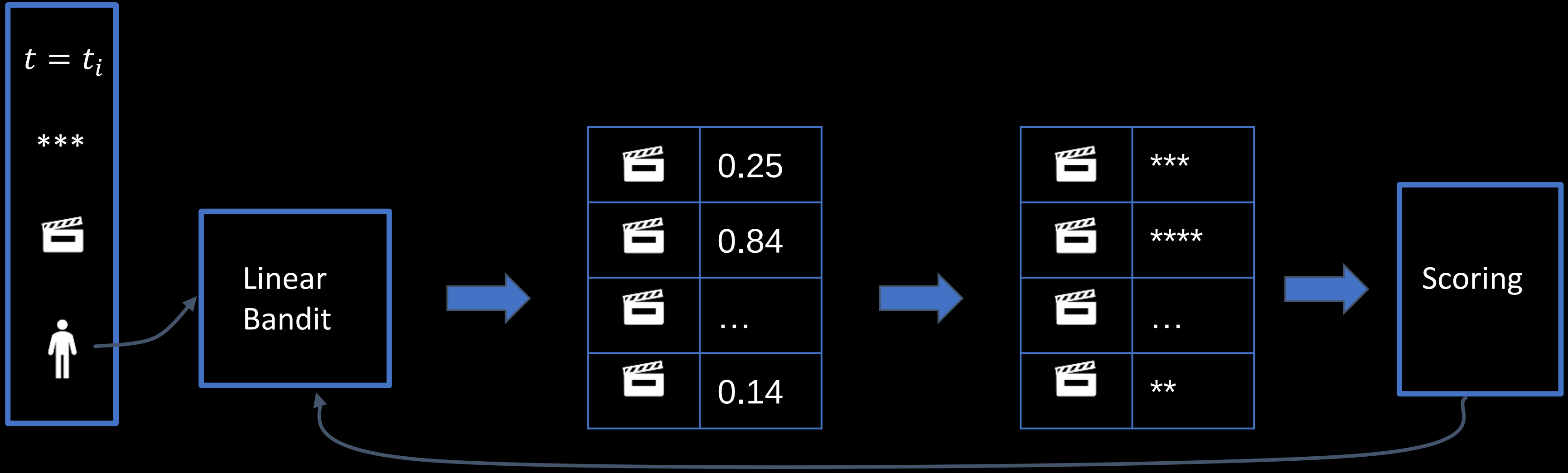
→ Conclusion: same performances as contextual linear bandits (PCA), but much higher runtime

→ Interpretation: user/movie interactions do not bring much information, users' features are not specific enough, same preferences among users?



Linear bandits in real time

At $t = t_i$: reception of a user → rating of all movies → comparison with reality, scoring and updating context



Linear bandits in real time

Pros and cons:

→ Less efficient in terms of calculation cost

→ Usable in real time

→ Not yet operational

For further research:

→ Repeat the experiment with a bigger data set

→ Collect precise users' features to bring proper context to the learning process

→ Take into account distance between users