Review of paper O. Chapelle, L. Li: "An empirical evaluation of Thomson sampling", (NIPS conference 2011, 9 citations in Scopus)

Mateusz Kobos

2016-12-13, ADA Lab seminar, ver.1

Summary

Thomson sampling is an old heuristic for addressing exploration/exploitation trade-off in learning systems, i.e. bandit problems. It's very effective, as the examples in the paper show, but unpopular in the literature.

Multi-armed bandit [Sect. 2]

They consider **contextual bandit** problem [Sect. 2]. This is defined as follows:

- We have:
  - context x (which is optional) - this is something that the agent is given from the environment
  - set of actions A - the agent can choose one of actions
  - after choosing an action $a \in A$, the agent observes a reward r.
- The goal: find a policy that selects actions such that the cumulative reward is as large as possible.

In case where x is missing, we have a problem of **multi-armed bandit problem**. In such case, we have a Las Vegas bandit machine with a number of arms. Our action is pulling the arm and this results in some kind of monetary reward (or lack of it).

Standard batch (off-line) predictive systems (classification and regression) can be applied here. In such setting: x is a feature vector, a is predicted class/value. After doing the prediction, the information about accuracy of the prediction is gathered and this corresponds to a reward. Having this information in hand, the system parameters are updated to make better predictions in the future.

Thompson sampling - idea [Sect. 2]

The idea of **Thompson sampling** is to randomly choose action according to its estimated probability of being optimal [Sect. 1].

Property of this approach:

- The more different are the actions explored, the better we can estimate their probability of being optimal. On the other hand, it would be best to always choose the single optimal action to get the best total reward.

Thompson sampling - multi-armed Bernoulli bandit [Sect. 2]

In **case of multi-armed Bernoulli bandit problem** [Algorithm 2] (where each arm has a certain probability of success assigned to it):

- For each arm we draw a random number from estimated distribution of reward for this arm (the estimated distribution is a Beta distribution).
- We pull the arm for which the largest number was drawn.

Thompson sampling - multi-armed Bernoulli bandit - example

Fig. 1: Thompson sampling for multi-armed Bernoulli bandit - example of updating the beta distribution estimates

Thompson sampling - generic algorithm [Sect. 2]

In general, the **Thompson sampling algorithm works like this** (my interpretation) [Algorithm 1]:

1. Retrieve current context x_t
2. Select action a_t randomly according to current model of the distribution of rewards $P(r|a, x, \phi)$; see Fig. 2 for a related illustration. In the paper they split this into two steps:
   - 2.1 Draw parameter(s) ϕ_t according to probability of this parameter taking into consideration data points so far $P(\phi|D)$.
   - 2.2 Select action a_t that maximizes the expected reward in the model of the distribution of rewards in which $\phi = \phi_t$.

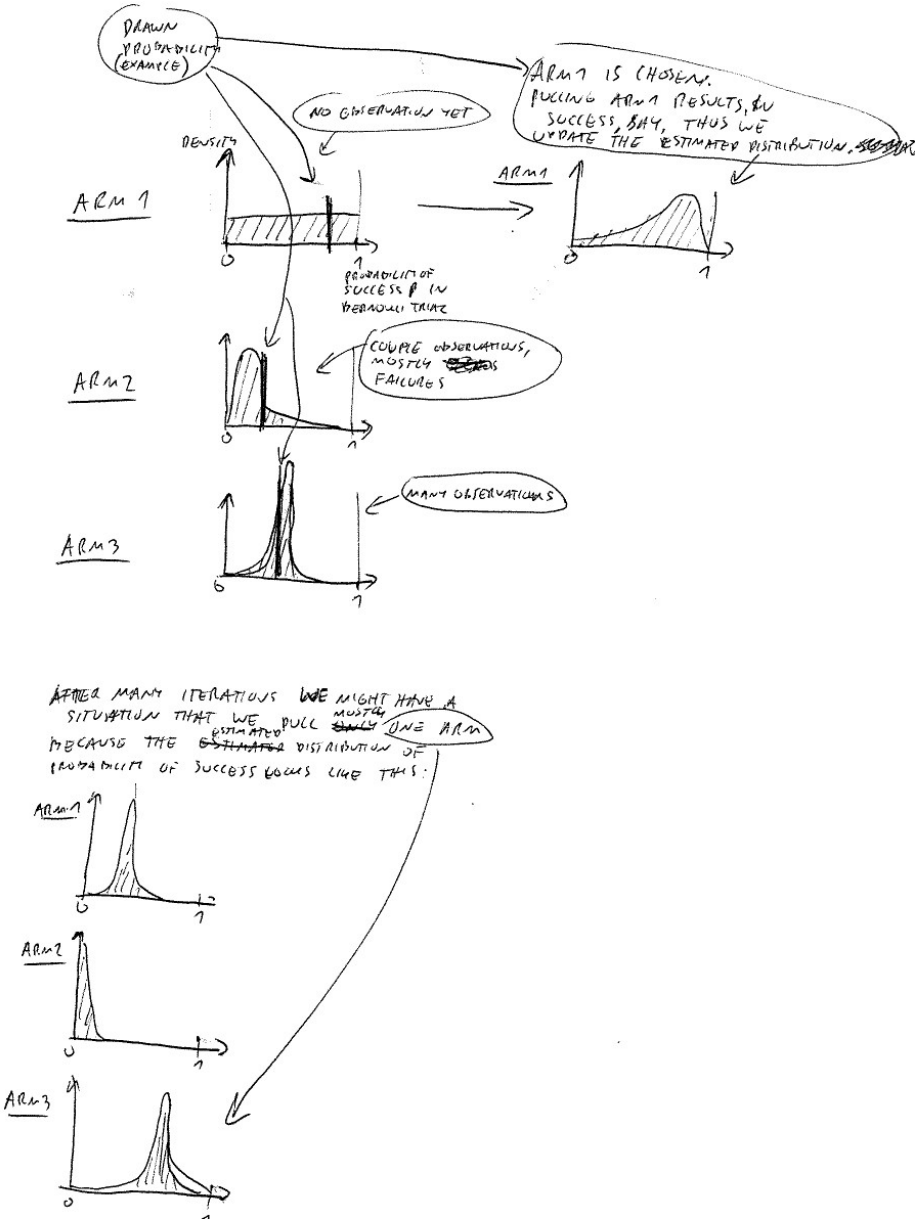
DRAWN PROBABILITY (EXAMPLE)
ARM 1 IS CHOSEN. PULLING ARM 1 RESULTS IN SUCCESS, SAY, THUS WE UPDATE THE ESTIMATED DISTRIBUTION.
NO OBSERVATION YET
DENSITY
ARM 1
ARM1
PROBABILITY OF SUCCESS p IN BERNOULLI TRIAL
ARM 2
COUPLE OBSERVATIONS, MOSTLY FAILURES
MANY OBSERVATIONS
ARM 3
AFTER MANY ITERATIONS WE MIGHT HAVE A SITUATION THAT WE PULL MOSTLY ONE ARM BECAUSE THE ESTIMATED DISTRIBUTION OF PROBABILITY OF SUCCESS LOOKS LIKE THIS:
ARM1
ARM2
ARM3

Figure 1:

3. Observe the reward r_t and update model's parameters ϕ by taking into the consideration new data point (x_t, a_t, r_t).

Fig. 2: Thompson sampling: the real distribution of rewards vs. the estimation (model).

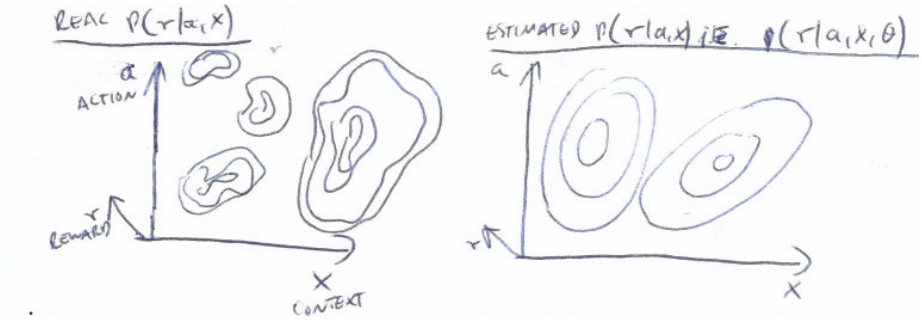

Figure 2:

Tidbits

- The standard way to estimate the probability of success (i.e. the reward) in Bernoulli distribution is to use **Beta distribution**. The Beta distribution is updated - usually becomes narrower - as more data about the results of Bernoulli trials become available. The initial Beta distribution used here is Beta(1, 1), which corresponds to uniform distribution on range $[0, 1]$. Beta distribution is a **conjugate distribution** of the binomial distribution (this is a term from Bayesian statistics).
- The quality of the algorithm is measured with **regret**. In case of multi-armed bandit problem, this is the difference between sum of rewards gathered by 1) an algorithm that always chooses the same arm which is the best on average and 2) the examined algorithm. [en.wikipedia.org/wiki/multi_amed_bandit]
- **Bayesian logistic regression** is used. In this approach, coefficients β are r.v.s with a certain prior distribution. The algorithm is updated when new data becomes available, as a result, it produces posterior distribution of β. One can calculate the "final" coefficient values from these distributions by selecting mean or mode or something similar. [en.wikipedia/wiki/linear_regresssion]

Example applications [Sects. 3, 4, 5]

Thompson sampling approach is tested on **couple example applications** - see below.

Simulation of a multi-armed bandit [Sect. 3]

Here one arm has a slightly higher probability of reward than the rest.

- No batch (off-line) prediction algorithm is used.
- The space that we explore/exploit: the arms to pull.

Display advertising [Sect. 4].

- The goal: choose which ad to show to the user browsing given webpage. For each `(user, webpage, time moment)` triple we have a different set of ads to choose from.
- Batch prediction algorithm used:
  - (Bayesian) logistic regression.
- Attributes: information about: webpage, user, and the ad to be served.
  - There are ~17M attributes and feature hashing is used.
- The space that we explore/exploit: weights of the regression model.
- How the algorithm works:
  1. Draw weights ϕ_t from posterior distribution of the weights from Bayesian logistic regression model.
  2. Apply the model with drawn weights ϕ_t to data at hand, i.e. to all ads; select the ad with the best score.
  3. Update the model [Algorithm 3], i.e. posterior distribution of weights, after obtaining a batch of feedback data points $(x_j, y_j),\ j = 1, \ldots, n$ where x_j are the values of input attributes and y_j is the information if the ad was clicked or not.
     - This is done by creating a new Bayesian logistic regression model on this batch of data.
     - The model uses regularization that punishes changes to the current model.
     - The obtained weight parameters are going to be used from now on.

News article recommendation [Sect. 5].

- The goal: choose which article from a pool of ~20 to show to the user. The articles in the pool are replaced with new ones from time to time.
- Batch prediction algorithm used:
  - (Bayesian) logistic regression.
  - Each article from the pool has its own Bayesian logistic regression model.
- Attributes: information about the user.
  - There are ~20 attributes, reduced from the original number of ~1000 attributes using PCA.
  - Features of articles were not used.
- The space that we explore/exploit: weights of the regression model for each of the articles from the pool. Another interpretation: we treat articles as arms in the multi-armed bandit [Sect. 5, paragraph 1].
- How the algorithm works: it works similarly to the one for display advertising. The differences:
  - We maintain a Bayesian logistic regression model for each article in the pool
  - We choose the article that results in the highest probability as predicted by individual logistic regression.

What things were tested [Sects. 3, 4, 5]

- Algorithms:
  - the basic Thompson sampling approach was compared with some **modifications of this approach** (depending on example:
    - **optimistic Thompson sampling** (we artificially increase the expected reward of actions)
    - **posterior reshaping** (there is a parameter α that manipulates exploration-exploitation trade-off))
  - and with alternative context bandit algorithms (depending on example:
    - **Upper Confidence Bound (UCB)** - a popular method with strong theoretical guarantees on the regret,
    - **exploit-only** (select the action with the highest mean - you select the best one w.r.t. current knowledge, you don't explore alternatives),
    - **random** (select the action uniformly at random),
    - **ϵ-greedy** (a mix between random and exploit-only - use the random approach with probability ϵ, exploit only otherwise)).
- Impact of delayed information about the results of actions was inspected.

Where the data for validating the model was taken from [Sects. 3, 4, 5]

Normally we don't know the reward of action that was not chosen, but we need to compute compute the effectiveness (regret) somehow. These are the approaches used:

- Simulation of a multi-armed bandit: all was simulated.
- Display advertising: the context was real, but the clicks were simulated using logistic regression using a modification of a weight vector learned on real clicks.
- News article recommendation: "replayer" approach, where a fraction of users is shown a totally (uniformly) random article.

Conclusions [Sect. 6]

Conclusions from the article:

- Thompson sampling is an effective heuristic for addressing exploration/exploitation trade-off.
- Using a version where the exploitation is boosted at the expense of exploration might be beneficial.
- The method is easy to implement and should be used as a baseline approach.
- The algorithm is robust to delayed feedback, unlike UCB that it was compared to.

My conclusions:

- (Bayesian) logistic regression is a popular model