Semi-parametric dynamic contextual pricing

Virag Shah — Jose Blanchet — Ramesh Johari
Stanford University
virag@stanford.edu

September 27, 2019

Dynamic pricing

- Several platforms have access to data describing history of different users.
- Platforms can leverage this information to price different products and optimize revenue.
- This requires learning online the mapping from user context to optimal price, efficiently.

Basic Framework

- Discrete times $1, 2, \ldots, n$, one user arrives per time step
- Each user is shown one product, which is ex-ante fixed
- Let V_t be the value t^{th} user assigns to the product.
- Let p_t be the price set by the platform.
- The user buys the product if $p_t \leq V_t$.
- Platform does not know or observe V_t, but has access to covariates $X_t \in \mathbb{R}^d$ which may describe user's history and product's type
- Goal: set prices $p_1, \ldots, p_n$ so as to maximize $\sum_{t=1}^{n} p_t \mathbf{1}\{p_t \leq V_t\}$.

Predicting V_t: The Data

- Input: $\{X_i, p_i\}_{i=1}^{t-1}$.
  X_t : covariate. p_t: price

- Output: $\{Y_i\}_{i=1}^{t-1}$, where

$$Y_i = \begin{cases} 1 \text{ if } V_i - p_i \geq 0 \\ 0 \text{ otherwise} \end{cases} .$$

  In other words, Y_i captures whether i^{th} was success or failure.

Predicting V_t: The Model

- Suppose,
$$V_i = \theta_0^\intercal X_i + Z_i,$$
θ_0: unknown parameters, Z_i: unknown demand-shock/noise.

- If we assume $Z_1, \ldots, Z_{t-1}$ are i.i.d. **Gaussian**, then we obtain **Probit model** with input $\{X_i, p_i\}_{i=1}^t$.

- Similarly, we obtain Logistic model for i.i.d. Logistic distributed demand-shocks.

Non parametric noise

- In this talk, we will not make any Gaussian or Logistic noise assumptions.

Non parametric noise

- In this talk, we will not make any Gaussian or Logistic noise assumptions.
- Even if we knew θ_0 exactly, the actual structure of distribution of demand-shocks is important to set optimal markups.

Non parametric noise

- In this talk, we will not make any Gaussian or Logistic noise assumptions.
- Even if we knew θ_0 exactly, the actual structure of distribution of demand-shocks is important to set optimal markups.
- Can we learn the actual distribution of shocks along with θ_0?

Predicting V_t: Our Semi-parametric Model

- Let
$$\ln V_i = \theta_0^\intercal X_i + Z_i,$$
θ_0: unknown parameters, Z_i: unknown demand-shock/noise.

- Demand-shocks are i.i.d. with arbitrary (non-parametric) distribution.

- Input: $\{X_i, p_i\}_{i=1}^{t-1}$, Output: $\{Y_i\}_{i=1}^{t-1}$ where $Y_i = \mathbf{1}\{V_i - p_i \geq 0\}$.

Semi-parametric Model: Curse of Dimensionality

- Usually, such semi-parametric models with binary feedback are hard to learn.
- That is one of the reasons why previous works on dynamic contextual pricing only consider probit/logistic/generalized-linear models.

Semi-parametric Model: Curse of Dimensionality

- Usually, such semi-parametric models with binary feedback are hard to learn.
- That is one of the reasons why previous works on dynamic contextual pricing only consider probit/logistic/generalized-linear models.
- Key leverage we find and use: Platform controls prices, and it is possible to focus learning in the regions which are revenue maximizing.

Price Experimentation

Conflicting goals:

- Observe X_t. Set price p_t to better learn θ_0 and distribution of Z. (exploration)
- Observe X_t. Set price p_t to maximize revenue. (exploitation)

Price Experimentation

Conflicting goals:

- Observe X_t. Set price p_t to better learn θ_0 and distribution of Z. (exploration)
- Observe X_t. Set price p_t to maximize revenue. (exploitation)

We view this as contextual bandit problem with prices as arms to maximize revenue.

Price Experimentation

Conflicting goals:

- Observe X_t. Set price p_t to better learn θ_0 and distribution of Z. (exploration)
- Observe X_t. Set price p_t to maximize revenue. (exploitation)

We view this as contextual bandit problem with prices as arms to maximize revenue.

As we learn more, we can zero in the further exploration near profit maximizing regions.

The Oracle

Let us first consider a simpler problem.

- Consider an Oracle which actually knows θ_0 and the distribution of Z.
- Now, let
$$F(z) = z\mathbb{P}(Z \geq \ln z), \text{ and } z^* = \arg\sup_z F(z).$$

Proposition

The following pricing policy maximizes revenue for the Oracle*: At each time t set price p_t^* such that*

$$\ln p_t^* = \theta_0^\intercal X_t + \ln z^*.$$

Regret

- Platform's revenue: $\Gamma_n = \sum_{t=1}^{n} p_t \mathbf{1}\{p_t \leq V_t\}$.
- Oracle's expected revenue: $\Gamma_n^* = \sum_{t=1}^{n} p_t^* \mathbf{1}\{p_t^* \leq V_t\}$.
- Expected regret: $\mathbb{E}[R_n] = \mathbb{E}[\Gamma_n^*] - \mathbb{E}[\Gamma_n]$.
- Maximizing $\mathbb{E}[\Gamma_n]$ is equivalent to minimizing $\mathbb{E}[R_n]$.
- We will focus on minimizing $\mathbb{E}[R_n]$, asymptotically in n.

Designing Optimal Algorithm: Key Ideas

- Recall, revenue maximizing policy for Oracle: $\ln p_t^* = \theta_0^\intercal X_t + \ln z^*$.
- For each z and θ, think of (z, θ) as an arm (i.e. a potential option). Pulling arm (z, θ) is equivalent to setting price p_t such that $\ln p_t = \theta^\intercal X_t + \ln z$.
- $(z, \theta) \in \mathbb{R}^{d+1}$: Curse of dimensionality?
- Important observation: Given X_t, for each choice of price p_t we *simultaneously* obtain information about the expected revenue for a *range* of pairs (z, θ).

DEEP-C Pricing Algorithm: Summary

DEEP-C: Dynamic Experimentation and Elimination of Prices - with Covariates.

- Maintain a set $A(t)$ of 'active arms' (z, θ) at each time.
- At time t, observe X_t and compute the set of active prices:

$$P(t) = \{p_t : \exists (z, \theta) \in A(t) \text{ s.t. } \ln p_t = \theta^\intercal X_t + \ln z\}.$$

- Choose price p_t at random from $P(t)$.
- Observe the revenue obtained. Eliminate (z, θ)'s from $A(t)$ for which there is enough information about sub-optimality.

The main result

Under some smoothness, compactness, independence, etc. assumptions, the following holds.

Theorem

The expected regret of DEEP-C algorithm satisfies the following: there exists a constant c such that

$$\mathbb{E}[R_n] = O\left(d^c\sqrt{n}\right).$$

Conclusions

- To learn via price experimentation, we do not need to make parametric (probit/logistic type) assumptions.
- We have a provably efficient algorithm which works under a 'very general' setting.