Структурированное ценообразование для взаимозаменяемых товаров

Н. К. Савельев[1], **Ю. В. Дорн**[2], **В. В. Стрижов**[3]

Аннотация: В работе исследуется задача поиска оптимальных, с точки зрения максимизации дохода, цен для группы взаимозаменяемых товаров (субститутов). Предлагается способ построения мультиагентной системы на основе исторических данных о покупках для моделирования распределения персональных уровней (private value) цен агентов. Преимуществами данной системы по сравнению с системами, моделирующими функцию спроса, является гибкость, возможность персонального ценообразования и более точное моделирование структуры связей между товарами. Разработан алгоритм дискретной оптимизации цен для произвольной выборки агентов. Для дальнейшего поиска оптимальных цен для всего множества агентов применяются алгоритмы обучения с подкреплением.

Ключевые слова: ценообразование; взаимозаменяемые товары; мультиагентная система; дискретная оптимизация; обучение с подкреплением.

[1]Московский физико-технический институт, savelev.nk@phystech.edu

[2]Московский физико-технический институт, dornyv@yandex.ru

[3]Вычислительный центр имени А. А. Дородницына Федерального исследовательского центра «Информатика и управление» Российской академии наук, Московский физико-технический институт, strijov@phystech.edu

Содержание

1 Введение

В работе рассматривается задача поиска оптимальных, с точки зрения максимизации дохода цен, для группы взаимозаменяемых товаров (субститутов). Для субститутов уществует прямое соотношение между ценой на один из них и спросом на другой, то есть снижение (повышение) цены одного товара вызывает уменьшение (увеличение) спроса на другой.

Существует множество работ, посвященных решению задачи ценообразования для одного товара или для группы независимых товаров. Товары независимы, если спрос на каждый из них не зависит от цен на другие. Особенностью данной задачи является то, что данных для ее решения нет или очень мало, а собирать эти данные, экспериментируя с установкой различных цен дорого, ведь если цены не оптимальны, то продавец теряет прибыль. Поэтому для решения задачи ценообразования, в основном, применяются методы обучения с подкреплением (активного обучения) и алгоритмы многоруких бандитов [5, 7], в том числе сэмплирование Томпсона [6] и предсказание на основе действий экспертов [3].

Пу сути, алгоритмы многоруких бандитов решают следующую задачу: дано множество действий $\mathcal{A} = \{a_1, \ldots, a_n\}$ и задана случайная функция награды $r : \mathcal{A} \to \mathbb{R}$, причем распределения наград для каждого из действий фиксированы и не меняются со временем. В каждый момент времени t агент совершает действие a_t и получает некоторую награду $r(a_t)$. Для фиксированного момента времени T требуется решить оптимизационную задачу

$$R(T) = \sum_{t=1}^{T} \mathbb{E}(r(a_t) - r(a^*)) \to \min, \tag{1}$$

где a^* - это действие, для которого матожидание награды $\mathbb{E}r(a^*)$ максимально. Функция $R(T)$ называется регретом и равна средним потерям от действий агента относительно оптимального действия за время T.

В работе [2] описано применение алгоритмов активного обучения для ценообразования в онлайн магазинах и на онлайн аукционах.

Формально, магазин отличается от аукциона тем, что во время аукциона известно, за какую цену покупатели готовы были приобрести тот или иной товар, а при продаже товаров в магазине эта информация не доступна продавцу. Алгоритмы, предложенные в работе [2], достигают оптимальных верхних и нижних оценок на регрет. Кроме того, авторами этой статьи предложен способ масштабирования цен на различные товары и способ добавления новых цен для их использования в качестве действий в многоруком бандите.

В случае, когда товары являются взаимозаменяемыми, нельзя искать оптимальные цены отдельно для каждого из них, так как установление оптимальной цены на один из товаров может обрушить спрос на другие. Поэтому действиями в многоруком бандите должны быть уже не цены на конкретный товар, а совокупность цен на всю группу взаимозаменяемых товаров. Таких совокупностей цен может быть огромное число (если имеется k возможных цен на n товаров, то их совокупностей k^n), поэтому применение алгоритмов многоруких бандитов без каких-либо доработок и изменений к задаче ценообразования для взаимозаменяемых товаров не представляется возможным.

Одним из способов решения задачи ценообразования для взаимозаменяемых товаров, встречающихся в литературе, является моделирование совместной функции спроса, аргументом этой функции является вектор цен на группу товаров, а значением - вектор спроса. Так, например, в работе [4] функция спроса моделирутся с помощью оценки вектора эластичностей цен для группы товаров. Спрос на отдельный i-й товар моделируется следующим образом:

$$d_i(x) \sim \left(\frac{x}{p_{0,i}}\right)^{\gamma_i}, \tag{2}$$

где $p_{0,i}$ - это текущая цена на i-й товар, а γ_i - компонента вектора эластичностей цен, относящаяся к i-му товару. Целевая функция дохода $Rev(p)$ есть сумма произведений цен на спрос для соответствующих товаров, оптимальные цены есть решение оптимизационной задачи:

$$Rev(p) = \sum_i p_i \cdot d_i(p_i) \to \max. \tag{3}$$

Алгоритм, предложенный авторами статьи [4], состоит из следующих шагов: сэмплирование вектора эластичностей цен γ из априорного распределения; моделирование спроса в соответствии с формулой (2); нахождение и установка оптимальных цен с точки зрения задачи (3); получение реального значения дохода и обновление априорного распределения эластичностей в соответствии с значениями реального и рассчитанного дохода. Таким образом, зависимость между товарами моделируется через зависимость компонент вектора элостичностей цен.

В данной работе предлагается уйти от моделирования совместной функции спроса и моделировать поведение покупателей. Каждый покупатель описывается вектором персональных уровней (private value) цен на некоторую группу товаров. Персональный уровень цены покупателя - это максимальное значение цены товара, при котором возможна покупка этого товара покупателем. Такой подход позволяет при произвольных ценах оценивать, какой из товаров предпочтет покупатель. То есть становится возможно моделирование выбора товара покупателем в зависимости от набора цен, установленных на группу товаров.

Другими словами, создается мультиагентная система, в которой каждый агент (покупатель) представляется одним вектором - вектором оценок персональных уровней цен, и на основе этого представления моделируется поведение агента, то есть его выбор товара при заданном векторе цен на всю группу товаров.

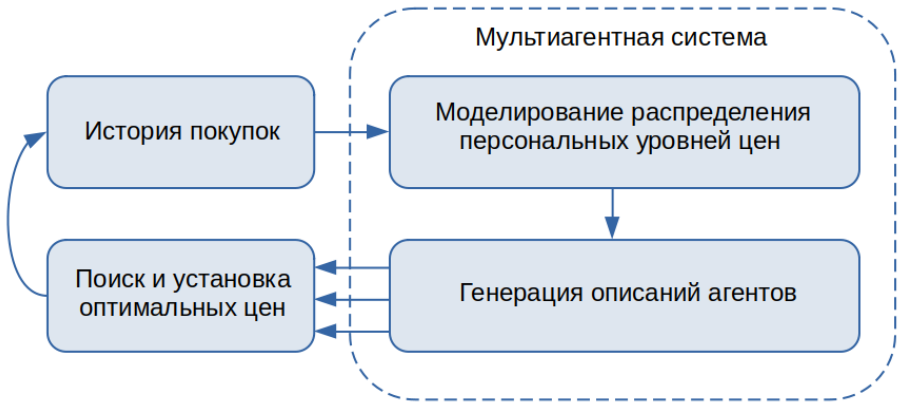


Рис. 1: Общая схема.

Предлагаемый подход моделирования поведения покупателей является более сложным, чем подход с моделированием функции спроса, но с другой стороны, он является более гибким, дает возможность персонального ценообразования, а также более чувствителен к структуре взаимосвязей между товарами.

2 Постановка задачи

Дано:

- $\mathcal{B}$ ($|\mathcal{B}| = k$) - множество взаимозаменяемых товаров;
- $\mathcal{C}$ ($|\mathcal{C}| = n$)- множество клиентов;
- $\{\mathcal{J}_t\}_{t=1}^{T}$ - история заказов, где $\mathcal{J}_t = \{(d_{i,j,t}, p_{i,t})\}_{i\in\mathcal{B},j\in\mathcal{C}}$, $d_{i,j,t}$ - количество товара i купленное пользователем j в момент времени t по цене $p_{i,t}$.

Фактический спрос $d_{i,j,t}$ при цене $p_{i,t}$ можно считать i-й компонентой реализации случайной многозначной функции спроса $d_j(p_t)$ клиента j, которая нам неизвестна.

Задача: Построить алгоритм ценообразования, максимизирующий ожидаемый доход за период времени T.

Описанная задача делится на несколько подзадач.

Первая из подзадач - это задача построения мультиагентной системы для моделирования агентов. Описанием агента является вектор размерности числа товаров k. Компоненты этого вектора - это персональные уровни цен агента на соответствующий товар. Описание агента формируется на основе исторических данных о его покупках. Задача заключается в разработке способа моделирования распределения персональных уровней цен агентов и и последующей генерации описаний агентов.

Вторая подзадача - это задача нахождения оптимального вектора цен для некоторого набора описаний агентов. Пусть $Q \in \mathbb{R}_+^{n\times k}$ - матрица описаний n агентов. Введем целевую функцию дохода $Rev : \mathbb{R}^k \to \mathbb{R}$,

$$Rev(p) = \sum_{j=1}^{k} p_j \sum_{i=1}^{n} \mathcal{I}(Q, p, i, j), \tag{4}$$

$$\mathcal{I}(Q, p, i, j) = \begin{cases} 1, \textit{if } \mathcal{S} := \{s : q_{is} \geq p_s\} \neq \emptyset \textit{ and } j = \underset{s\in\mathcal{S}}{\operatorname{argmax}}\, q_{is} \\ 0, \textit{otherwise} \end{cases}.$$

$\mathcal{I}(Q, p, i, j)$ - это индикаторная функция, она равна 1 в случае, когда i-й моделируемый агент выбирает для покупки j-й товар про фиксированном векторе цен p.

Тогда искомый вектор цен является решением оптимизационной задачи

$$Rev(p) \to \max_{p}. \tag{5}$$

Целевая функция $Rev(p)$ является негладкой, невыпуклой и мультимодальной, поэтому оптимизационная задача (5) сводится к задаче дискретной оптимизации, которая, в свою очередь, может быть решена точно за разумное время при ограничениях на число описаний агентов в исследуемом наборе.

Третья подзадача - это задача нахождения оптимального вектора цен всех описаний агентов из множества $\mathcal{C}$. Решением второй подзадачи является точный вектор оптисмальных цен для некоторого подмножества агентов, из-за сложности решения оптимизационной задачи (5) размер этого подмножества ограничен. Предлагается найти несколько оптимальных векторов цен для различных подмножеств агентов, затем использовать алгоритмы многоруких бандитов для динамического выявления лучшего из найденных векторов цен. Здесь уже возможно применение алгоритмов многоруких бандитов, так как действиями будут конкретные ветора цен, оптимальных для подмножеств агентов, и поэтому число действий будет небольшим, относительно случая с применением алгоритмов многоруких бандитов напрямую, когда действиями были бы всевозможные совокупности цен на множество товаров $\mathcal{B}$.

3 Построение мультиагентной системы

В этой главе рассматривается задача построения мультиагентной системы для моделирования поведения покупателей. Для этого предлагается способ моделирования распределения персональных уровней цен на множество взаимозамениямых товаров на основе исторических данных о покупках этих товаров и последующей генерации описаний агентов с помощью смоделированного распределения.

Большая часть клиентов имеют историю покупок только небольшой части товаров из исследуемой группы взаимозаменяемых товаров. Поэтому при генерации описания агента предлагается использовать оценки персональных уровней цен для части товаров, полученные с помощью известной истории покупок некоторого клиента. Таким образом формируется неполное описание агента, то есть вектор описания, содержащий пропуски (неизвестные компоненты). Затем вектор неполного описания агента дополняется до полного описания с помощью смоделированного распределения персональных уровней цен.

В основе мультиагентной системы лежит модель Conditional WGAN (Wasserstein Generative Adversarial Network). Ниобходимость использования расстояния Вассерштейна обоснована тем, что распределение персональных уровней цен агентов мультимодально, причем моды выражены сильно. Поэтому во время обучения распределения, оцениваемые в генераторе и дискриминаторе, могут слабо пересекаться, и для того, чтобы ускорить и сделать возможным процесс обучения GAN, необходимо использовать расстояние Вассерштейна. В дополнение к этому, используется именно условный (conditional) GAN для того чтобы учесть информацию из истории покупок заданного клиента, как было описано ранее.

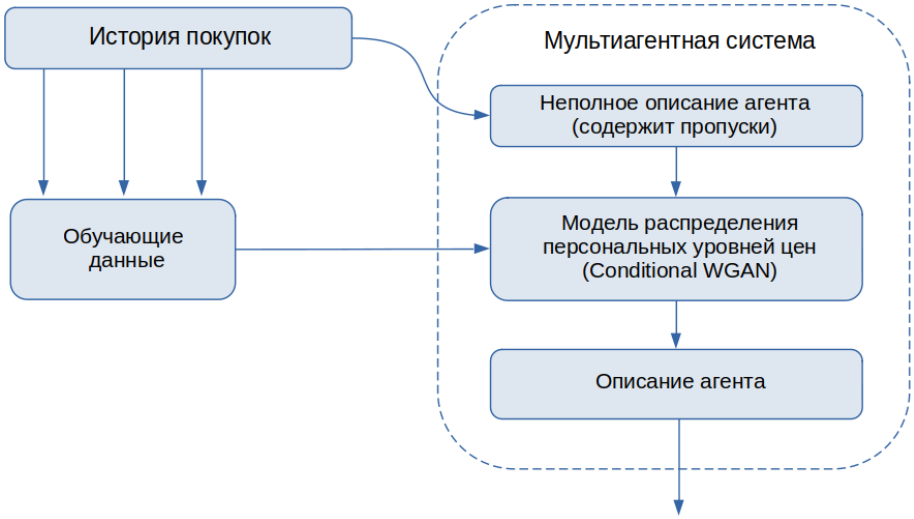


Рис. 2: Мультиагентная система.

Построение архитектуры Conditional WGAN было основано на работе [8]. Модель Conditional WGAN состоит из двух нейронных сетей - генератора и дискриминатора. Задача генератора - смоделировать реальное распределение персональных уровней цен агентов. Дискриминатор, в свою очередь, пытается отличить описания агентов, сгенерированные генератором, от настоящих описаний и выступает в качестве функции потерь для генератора.

Генератор состоит из двух блоков, содержащих линейный слой, батч-нормализацию и активацию ReLU, и выходного линейного слоя. Также в генераторе используются пробросы входа сети в глубокие блоки (residual connections) для улучшения сохранения градиента во время обратного распространения ошибки. Входом генератора является результат конкатинации двух векторов - условного вектора $\hat{q}$ и вектора скрытых переменных z.

Условный вектор - это вектор, содержащий известную информацию, об описанни агента, которое должна сгенерировать сеть. В качестве условного вектора используется вектор неполного описания агента, построенный на основе истории покупок. Неполное описание агента - это вектор $\hat{q} \in \mathbb{R}^k$ с компонентами:

$$\hat{q}_i = \max\{p_{i,t} : (d_{i,t}, p_{i,t}) \in \{\mathcal{J}_t\}_{t=1}^{T}\}, \quad \max\{\varnothing\} = Null, \tag{6}$$

где Null - специальное значение, обозначающее пропуск.

Вектор скрытых переменных генерируется случайным образом из многомерного нормального распределения. Значение вектора скрытых переменных влияет на дополнение пропусков в неполном описании агента с помощью моделируемого генератором распределения персональных уровней цен. Дополненный вектор описания агента является выходом генератора.

Дискриминатор, в свою очередь, состоит из двух блоков, содержащих линейный слой, активацию LeakyReLU и слой Dropout, и выходного линейного слоя. Входом дискриминатора является выход генератора, а выходом - одно число от 0 до 1 - вероятность того, что вход сгенерирован из истинного распределения персональных уровней цен.

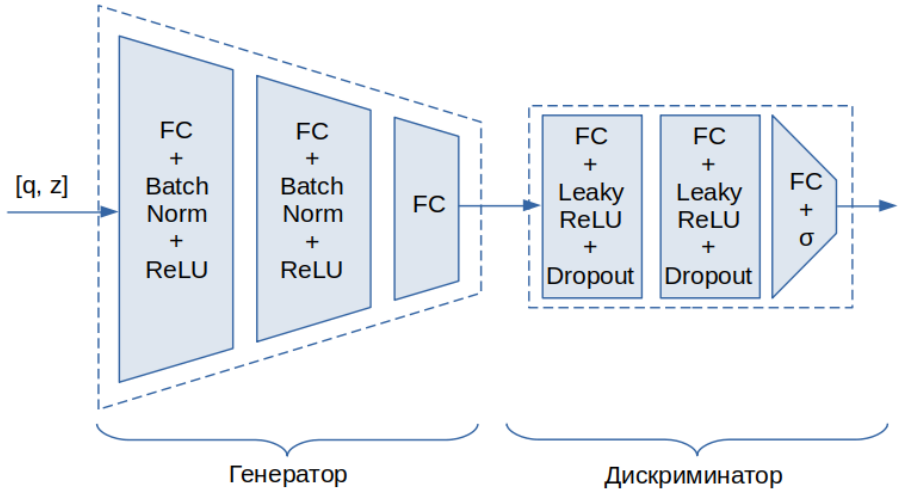


Рис. 3: Conditional WGAN.

Процесс обучения WGAN подробно описан в работе [1]. В качестве функции ошибки дискриминатора использовалась кроссэнтропия. Функцией ошибки генератора являлась сумма выхода дискриминатора и L1-норма разности условного вектора и выхода генератора, в котором обнулены компоненты, соответствующие неизвестным компонентам условного вектора. Шаг градиентного спуска для дискриминатора производился на каждой итерации обучения, а для генератора - раз в пять итераций. Для обучения Conditional WGAN

использовались неполные описания агентов, полученные в различные моменты времени истории покупок. Тем самым сеть была обучена дополнять недостающие персональные уровни цен в неполном описании агента.

После того, как модель Conditional WGAN обучена, процесс генерации описаний агентов в мультиагентной системе сводится к выбору некоторого клиента из множества $\mathcal{C}$, генерации неполного описания агента на основе истории покупок выбранного клиента в соответствии с формулой (6) и дополнения этого описания с помощью обученного генератора Conditional WGAN.

4 Поиск оптимального вектора цен

В этой главе рассматривается задача поиска оптимального с точки зрения максимизации дохода вектора чен на множество товаров $\mathcal{B}$ для некоторого набора описаний агентов.

Пусть $Q \in \mathbb{R}_+^{n \times k}$ - матрица описаний n агентов. Введем целевую функцию дохода $Rev : \mathbb{R}^k \to \mathbb{R}$,

$$Rev(p) = \sum_{j=1}^{k} p_j \sum_{i=1}^{n} \mathcal{I}(Q, p, i, j), \tag{7}$$

$$\mathcal{I}(Q, p, i, j) = \begin{cases} 1, \, if \, \mathcal{S} := \{s : q_{is} \geq p_s\} \neq \emptyset \; and \; j = \underset{s \in \mathcal{S}}{\operatorname{argmax}} \, q_{is} \\ 0, \, otherwise \end{cases}.$$

Здесь также введена индикаторная функция $\mathcal{I}(Q, p, i, j) : \mathbb{R}_+^{n \times k} \times \mathbb{R}^k \times \{1, \ldots, n\} \times \{1, \ldots, k\} \to \{0, 1\}$, она равна единице в случае, когда i-й агент выбирает для покупки j-й товар про фиксированном векторе цен p, и нулю в противном случае.

Тогда искомый вектор цен является решением оптимизационной задачи

$$Rev(p) \to \max_p . \tag{8}$$

Целевая функция $Rev(p)$ является негладкой, невыпуклой и мультимодальной, и поэтому в общем случае оптимизационная задача (8) не разрешима. Но в данном случае, благодаря свойствам рассматриваемой функции $Rev(p)$, можно свести оптимизационную задачу (8) к задаче дискретной оптимизации.

Утверждение. Для любой матрицы $Q \in \mathbb{R}_+^{n \times k}$, найдется такой набор индексов $i_1, \ldots i_k$, что вектор $(q_{i_1 1}, \ldots, q_{i_k k})^T$ является решением оптимизационной задачи (8).

Доказательство. Введем множества

$$Q_j = \{q_{ij}\}_{i=1}^{n}, \; j = 1, \ldots, k$$

и множество

$$P = \{p \in \mathbb{R}^k : \min Q_j \leq p_j \leq \max Q_j, j = 1, \ldots, k\}.$$

Тогда $\sup Rev(p) = \sup\limits_{p \in P} Rev(p)$. Рассмотрим произвольный вектор $x \in P$. Постором для него вектор

$$y : y_j = \min_{q \in Q_j, q \geq x_j} q.$$

Тогда $Rev(x) \leq Rev(y)$. Следовательно, можно искать максимум $Rev(p)$ на конечном множестве точек вида $(q_{i_1 1}, \ldots, q_{i_k k})^T$.

□

В силу утверждения выше, можно искать решение оптимизационно задачи (8) среди конечного множества векторов, составленных из элементов столбцов матрицы Q. Таким образом, оптимизационная задача (8) является задачей дискретной оптимизации.

Полученная задача дискретной оптимизации все еще очень сложна, для ее упрощения была разработана процедура под названием clean.

Пусть p^* - искомое решение оптимизационной задачи (8). Рассмотрим алгоритм процедуры clean, позволяющий значительно уменьшить мощность множества векторов, среди которых производится поиск решения p^*.

Algorithm 1 Процедура clean.

1. Упорядочим элементы матрицы Q по убыванию.
2. Рассмотрим первый элемент получившегося упорядоченного набора - $q_{i_1 j_1}$.
   $p^*_{j_1} \leq q_{i_1 j_1} \Rightarrow I(Q, p, i_1, j) = 0,\ j \neq j_1\ (q_{i_1 j} \leq q_{i_1 j_1}) \Rightarrow$ можно убрать из рассмотрения все элементы вида $q_{i_1 j} : q_{i_1 j} \leq q_{i_1 j_1}$.
3. Рассматриваем следующий элемент обновленного упорядоченного набора, имеющий другой второй индекс - $q_{i_2 j_2}$, первый индекс этого элемента тоже будет другой, так как иначе он был бы удален на предыдущем шаге.
   $p^*_{j_2} \leq q_{i_2 j_2} \Rightarrow I(Q, p, i_2, j) = 0,\ q_{i_2 j} \leq q_{i_2 j_2} \Rightarrow$ можно убрать из рассмотрения все элементы вида $q_{i_2 j} : q_{i_2 j} \leq q_{i_2 j_2}$.
4. Аналогично повторяем шаг 3 для остальных индексов.

Конструктивное постороение алгоритма процедуры clean доказывает корректность ее применения для упрощения оптимизационной задачи (8).

Теорема (Савельев, 2021). Пусть $Q \in \mathbb{R}_+^{n \times k}, n \geq k$. Тогда функция

$$Rev(p) = \sum_{j=1}^{k} p_j \sum_{i=1}^{n} I(Q, p, i, j),$$

$$I(Q, p, i, j) = \begin{cases} 1, \textit{if } S := \{s : q_{is} \geq p_s\} \neq \emptyset \textit{ and } j = arg \max\limits_{s \in S} q_{is} \\ 0, \textit{otherwise} \end{cases}$$

достигает максимума на векторе вида $(q_{i_1 1}, \ldots, q_{i_k k})^T$, где q_{ij} - это элемент матрицы Q, который не был удален процедурой clean.

Следствие (Необходимое условие единственности максимума). Пусть $Q \in \mathbb{R}_+^{n \times k}$. Для того, чтобы точка максимума целевой функции $Rev(p)$ была единственна, необходимо выполнение неравенства $n \geq k$.

Доказательство. Пусть $n < k$. Применим к матрице Q процеду-

ру clean. На каждом шаге процедуры clean рассматривается элемент матрицы Q с индексами, отличными от индексов элементов, рассмотренных на предыдущих шагах. Следовательно, будет сделано n шагов. Но тогда в матрице Q будет столбец, ни один элемент которого не рассматривался в ходе процедуры clean. А значит, все элементы этого столбца меньше рассмотренных и все они будут удалены. Обозначим индекс этого столбца, как $\hat{j}$. Пусть p^* - некоторая точка максимума функции $Rev(p)$. Тогда изменение компоненты $p^*_{\hat{j}}$ не изменяет значения функции. Следовательно, точка максимума не единственна.

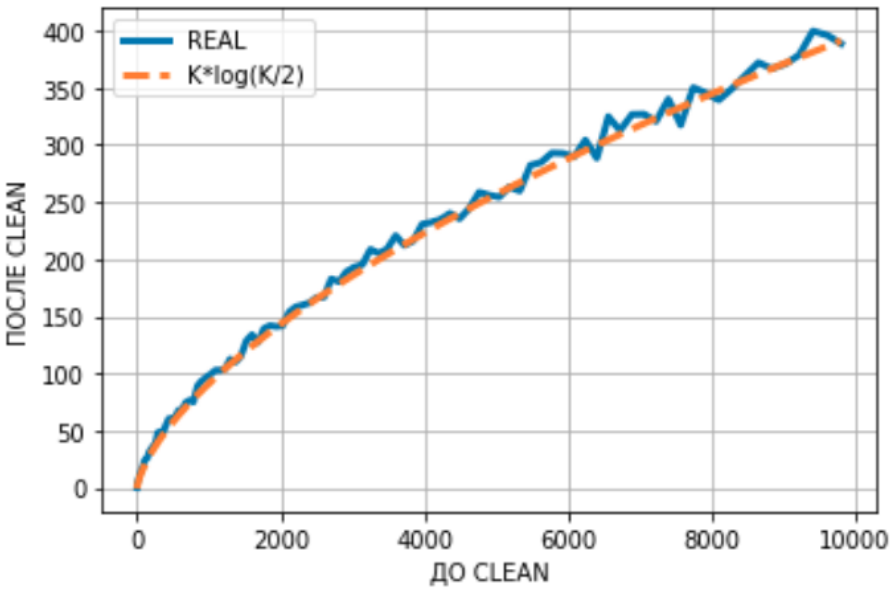
□

5 Экспериментальное исследование процедуры clean

В этой главе описано проведение экспериментов, направленных на исследование процедуры clean, и результаты этих экспериментов.

Первым этапом практичиских экспериментов стало исследование числа элементов, остающихся в матрице описаний агентов после применения к ней процедуры clean.

В ходе эксперимента было сгенерированно по двадцать квадратных матриц описаний агентов размерами от 1 до 100. Элементы матриц являлись независимыми равномерно распределенными на отрезке $[0, 1]$ случайными величинами. С каждой матрицей была произведена процедура clean. Затем для матриц фиксированного размера было посчитано среднее число элементов, оставшихся в матрице после применения к ней процедуры clean. Зависимость количества оставшихся элементов матрицы от изначального количества ее элементов представлена на графике.

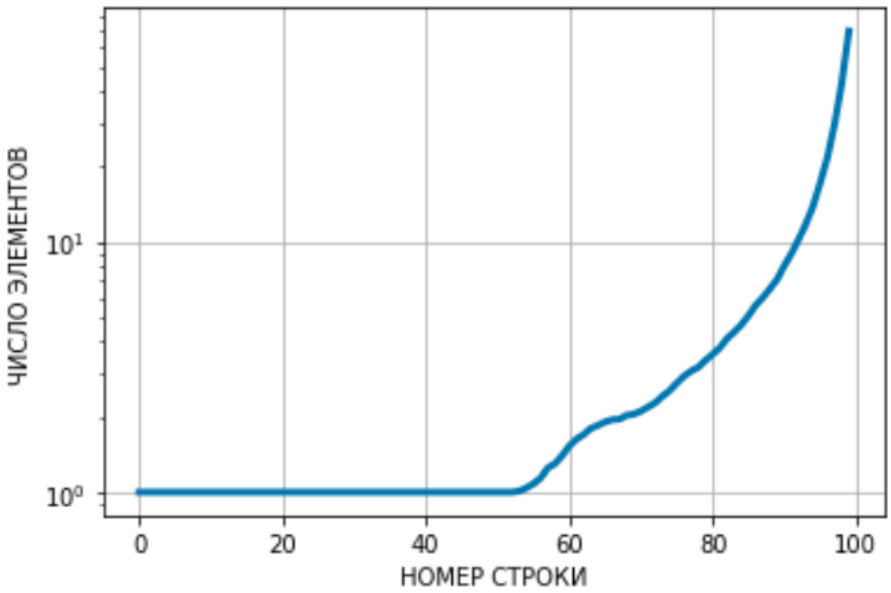


Эксперементально был установлен вид зависимости количества оставшихся элементов матрицы от изначального количества ее элементов. Если изначально в матрице было k^2 элементов, то после процедуры clean в среднем остается только $k \log(k/2)$ элементов. Таким

образом, при больших значениях k матрица теряет огромную часть своих элементов, что упрощает задачу поиска оптимального вектора цен.

Вторым этапом практичиских экспериментов стало исследование числа элементов, остающихся в отдельных строках матрице описаний агентов после применения к ней процедуры clean.

В ходе эксперимента было сгенерированно 100 квадратных матриц описаний агентов размерам 100 на 100. Элементы матриц являлись независимыми равномерно распределенными на отрезке $[0, 1]$ случайными величинами. С каждой матрицей была произведена процедура clean. Далее строки матриц были отсортированы по числу оставшихся в них элементов. Зависимость среднего числа оставшихся после процедуры clean элементов строк матриц описаний агентов от порядкого номера строки после сортировки представлено на графике.



Были получены довольно интересные результаты. Зависимость числа остающихся в строке элементов от номера строки довольно необычна и интересна. Но главное, что можно заметить - это то, что более половины строк матрицы после процедуры clean содержать лишь один элемент. А это значит, что с легкостью можно определить более половины координат искомого оптимального вектора цен, что

значительно облегчает задачу максимизации целевой функции дохода $Rev(p)$.

6 Заключение

В работе рассматривалась задача оптимизации поиска оптимальных с точки зрения максимизации дохода цен на группу взаимозаменяемых товаров.

Был предложен способ построения мультиагентной системы на основе исторических данных о покупках для моделирования поведения клиентов. Преимуществами разработанной мультиагентной системы являются возможность персонального ценообразования, гибкость и более точное моделирование структуры связей между товарами по сравнению с подходами, моделирующими совместную функцию спроса.

Также была введена целевая функция дохода и сформулирована оптимизационная задача, решением которой является вектор цен, максимизирующий доход от продажи товарав некоторому множеству агентов. Полученная задача была сведена к задаче дискретной оптимизации.

Разработана процедура clean, значительно упрощающая решение задачи дискретной оптимизации. Сформулирована и доказана теорема о корректности применения процедуры clean. В качестве следствия из теоремы выведено необходимое условие единственности решения оптимизационной задачи.

Проведено экспериментальное исследование процедуры clean. Эксперименты подтверждают эффективность ее применения для упрощения задачи дискретной оптимизации.

Список литературы

[1] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 214–223. PMLR, 06–11 Aug 2017.

[2] Sébastien Bubeck, Nikhil R. Devanur, Zhiyi Huang, and Rad Niazadeh. Online auctions and multi-scale online learning. *CoRR*, abs/1705.09700, 2017.

[3] N. Cesa-Bianchi and G. Lugosi. *Prediction, Learning, and Games*. Cambridge University Press, 2006.

[4] Ravi Ganti, Matyas Sustik, Quoc Tran, and Brian Seaman. Thompson sampling for dynamic pricing, 2018.

[5] T. Lattimore and C. Szepesvári. *Bandit Algorithms*. Cambridge University Press, 2020.

[6] Daniel Russo, Benjamin Van Roy, Abbas Kazerouni, and Ian Osband. A tutorial on thompson sampling. *CoRR*, abs/1707.02038, 2017.

[7] Aleksandrs Slivkins. Introduction to multi-armed bandits. *CoRR*, abs/1904.07272, 2019.

[8] Lei Xu, Maria Skoularidou, Alfredo Cuesta-Infante, and Kalyan Veeramachaneni. Modeling tabular data using conditional GAN. *CoRR*, abs/1907.00503, 2019.