

## 7. Динамический случай

# Формальная постановка задачи

Максимизировать прибыль  $y(p, \xi)$  по цене  $p \in [p, +\infty)$  .

4 разные постановки:

1.  $p^* = \operatorname{argmax}_p E[y(p_n, \xi)]$  за  $n$  итераций.
2.  $p^* = \operatorname{argmax}_p E[y(p, \xi)]$  за минимальное время.
3.  $\max_p \sum E[y(p, \xi)]$  Оптимизация траектории.
4. Найти зависимость  $y(p)$

$E[\cdot]$  - может быть  
любым оператором.  
Можно считать, как  
мат. ожидание.

# Что мы научились делать?



Мы научились решать задачу ценообразования в статическом(!), многомерном случае.

А где же динамика?

# Динамика



Мы неявно предполагали, что распределение  $y(p, \xi)$  никак не зависит от времени, и мы можем искать оптимум (или итерировать) «сколько нужно».

Но это очень серьезное допущение.

# Формальная постановка задачи

Максимизировать прибыль  $y_t(p, \xi)$  по цене  $p \in [p, +\infty)$ .

4 разные постановки:

1.  $p^* = \operatorname{argmax}_p E[y_t(p_n, \xi)]$  за  $n$  итераций.
2.  $p^* = \operatorname{argmax}_p E[y_t(p, \xi)]$  за минимальное время.
3.  $\max_p \sum E[y_t(p, \xi)]$  Оптимизация траектории.
4. Найти зависимость  $y_t(p)$

$E[\cdot]$  - может быть  
любым оператором.  
Можно считать, как  
мат. ожидание.

# Формальная постановка задачи

Какие предположения о  $y_t(m, \xi)$  можно сделать?

1. Динамика по  $t$  известна.
2. Оптимум по  $m$  не меняется со временем. Когда такое может быть?
3. Характер параметрической функции сохраняется со временем.
4.  $y_t(m, \xi)$  «не сильно» меняется со временем. «Не сильно» это как?

# Динамика по $t$ известна.

$$y_t(m, \xi) = z(t, y(m, \xi))$$

Где  $z(t, y)$  - известная функция.

Что это значит точки зрения поиска оптимума?

Если  $z(t, y)$  - возрастает по  $y$  – то

$$\operatorname{argmax}_m y_t(m, \xi) = \operatorname{argmax}_m y(m, \xi)$$

Если  $z(t, y)$  - убывает по  $y$  – то

$$\operatorname{argmax}_m y_t(m, \xi) = \operatorname{argmin}_m y(m, \xi)$$

# Оптимум по $m$ не меняется со временем



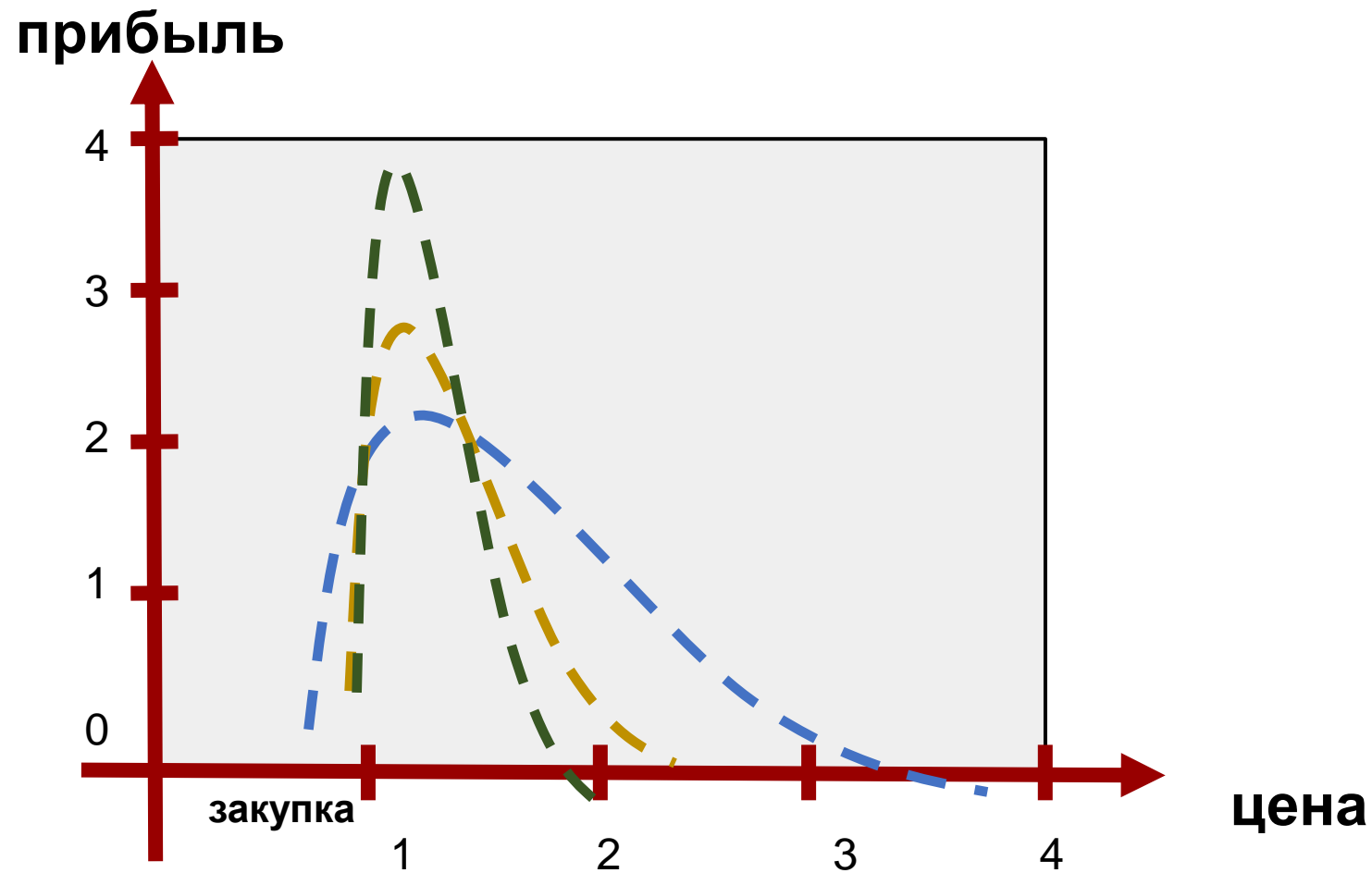
Что это значит?

Это означает, что рынок, полон информацией, с практически равными издержками и рынок эффективен. Самые большие объемы торгуются в районе оптимума.

Разумеется, существуют сделки, которые находятся в неэффективной зоне.



# Задача ценообразования



# Оптимум по $m$ не меняется со временем



Что это значит?

Это означает, что рынок, полон информацией, с практически равными издержками и рынок эффективен. Самые большие объемы торгуются в районе оптимума.

Разумеется, существуют сделки, которые находятся в неэффективной зоне.

На практике, чаще всего «коридор» сужается.

# Характер параметр. сохраняется

$$y_t(m, \xi) = Q_t(m)m,$$

- Линейная функция:  $Q_t(x) = \max(-a(t)x + b(t), 0)$
- Гиперболическая функция:  $Q_t(x) = \max(-a(t)/x + b(t), 0)$
- Экспоненциальная функция:  $Q_t(x) = \max(-\exp(a(t)x + b(t)) + c(t), 0)$
- Показательная функция:  $Q_t(x) = \max(a(t)x^{b(t)} + c(t), 0)$

Как решать такую задачу?

# Характер параметр. сохраняется

Если **раньше** (когда не было динамики по  $t$ ) мы с каждой итерацией оценивали вектор оптимальных параметров, полагая, что получаем «реализации».

$a_1, a_2, a_3, \dots$

$b_1, b_2, b_3, \dots$

Важно понимать, что на самом деле мы получаем не реализации, этих параметров. А мы получаем **реализацию случайной величины**, которая порождает оценки (в рамках наших допущений о параметризации) параметров.

Важно также, что если  $\widehat{a_{n-1}} < \widehat{a_n}$ . Чем позже мы оцениваем, тем точнее оценка, т.к. выборка становится больше.

# Характер параметр. сохраняется

Когда есть динамика по  $t$  – это действительно можно воспринимать как реализацию.

$a_1, a_2, a_3, \dots$

$b_1, b_2, b_3, \dots$

Что это означает?  $a$  как-то зависит от  $t$ . Но мы не знаем как. Если знаем, то см. пункт 1.

Даже если этой зависимости нет на самом деле, а мы ее предполагаем, т.е.  $a(t) = a$ , то в общем случае  $\widehat{a_{n-1}} < \widehat{a_n}$  неверно! **И это проблема.**

# Характер параметр. сохраняется

Что с этим делать?

Находить зависимость  $a(t)$  - как в классической задаче регрессии.

На практике, как правило, в динамике по  $t$  есть тоже свои особенности:

- Сезонность глобальная
- Сезонность локальная (выходные-будни)
- Тренд

См. анализ временных рядов.

# Характер параметр. сохраняется

Есть реализации  $\xi_1, \xi_2, \xi_3 \dots$  Она порождает:

$$y_1(m) = Q_{t=1}(m)m \rightarrow a_1, b_1 \dots$$

$$y_2(m) = Q_{t=2}(m)m \rightarrow a_2, b_2 \dots$$

$$y_3(m) = Q_{t=3}(m)m \rightarrow a_3, b_3 \dots$$

Важно, что мы не агрегируем выборку, а считаем, что это независимые наблюдения по  $t$ .

- Для решения проблемы «старта». На первых нескольких итерациях считаем, что  $\mathbf{a}, \mathbf{b}$  не зависит от  $t$ . (статическая часть)
- Повторять в цикле (динамическая часть):
  - После этого решаем задачу регрессии  $(t, \mathbf{a}, \mathbf{b})$ . Восстанавливаем  $\mathbf{a}(t), \mathbf{b}(t)$ .
  - При поиске оптимального  $m$  на следующем шаге подставляем в  $\mathbf{a} = \mathbf{a}(t+1)$  в функцию  $Q$ .

# «Не сильно» меняется.



Довольно очевидно, что если  $y_t(m)$  имеет сложную (не предсказуемую) динамику. Например, сложность в системе растет быстрее чем выборка, или по какой-то причине качество ее прогнозирования низкое, то задача не имеет большого смысла.

Нужны какие-то ограничения на динамику.

Можно использовать следующие ограничения:

$$y_t(m)c_1 \leq y_{t+1}(m) \leq y_t(m)c_2,$$

где коэффициенты задают некоторый «коридор».



# «Не сильно» меняется.



Или в терминах параметрической функции:

$$a(t)c_1 \leq a(t+1) \leq a(t)c_2,$$

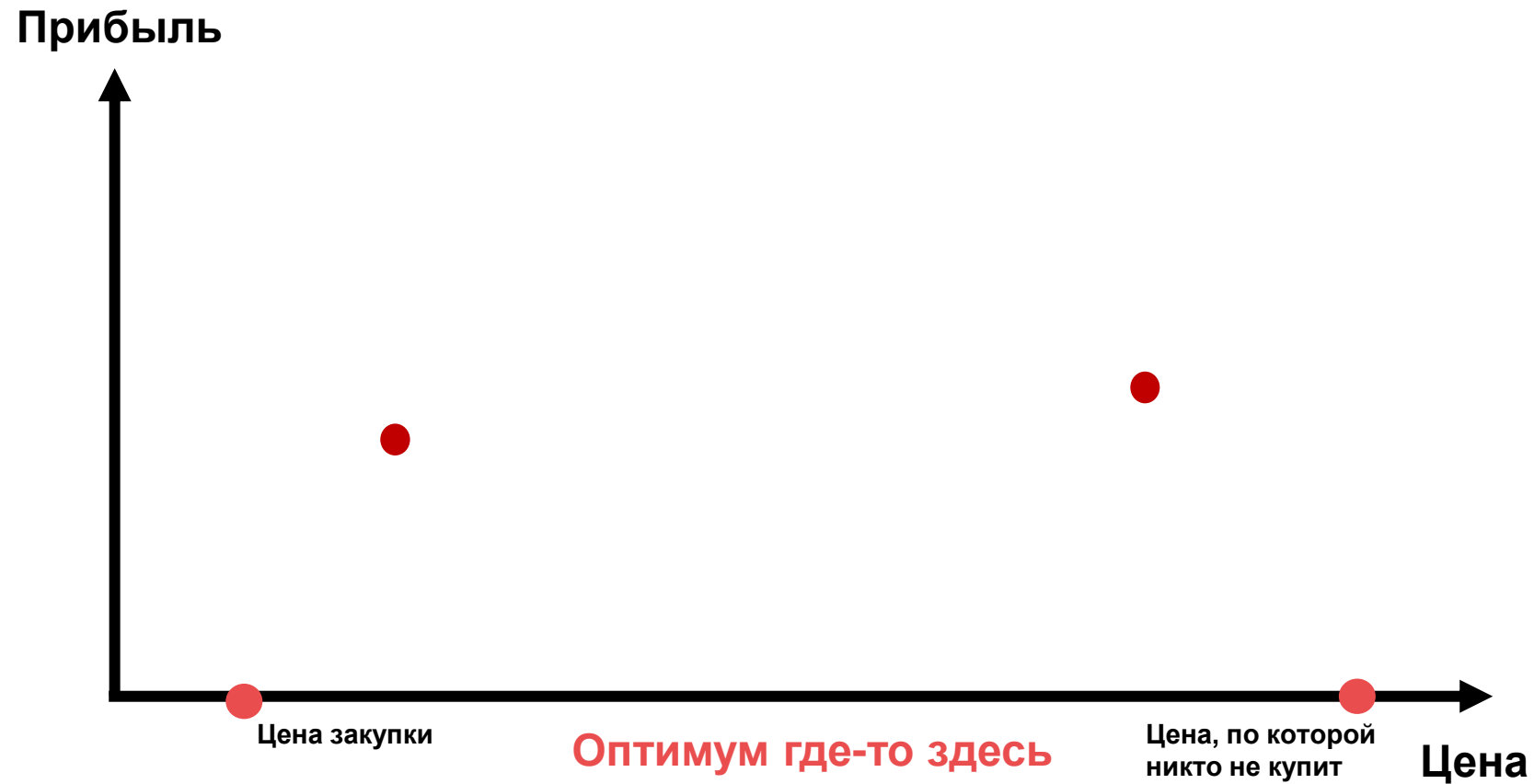
где коэффициенты задают некоторый «коридор».

Разумеется «коридоры» могут быть разные.

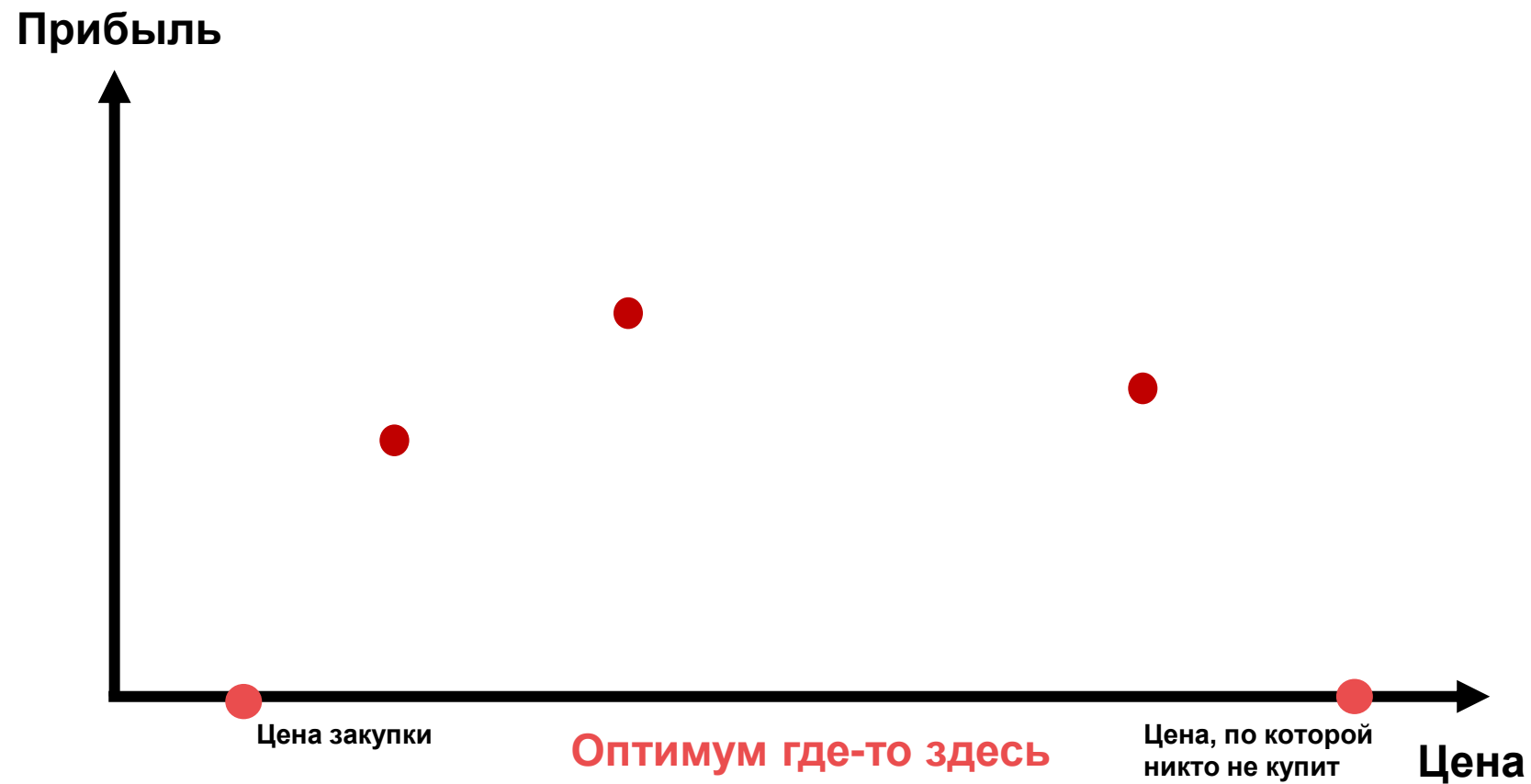


Общая «нормировка»

# Задача ценообразования



# Задача ценообразования



# Задача ценообразования





# «Нормировка»

Вообще говоря, если мы знаем динамику спроса по  $t$  «при прочих равных», то мы можем слегка модифицировать задачу.

$$y_t(m, \xi) = Q_t(m)m,$$

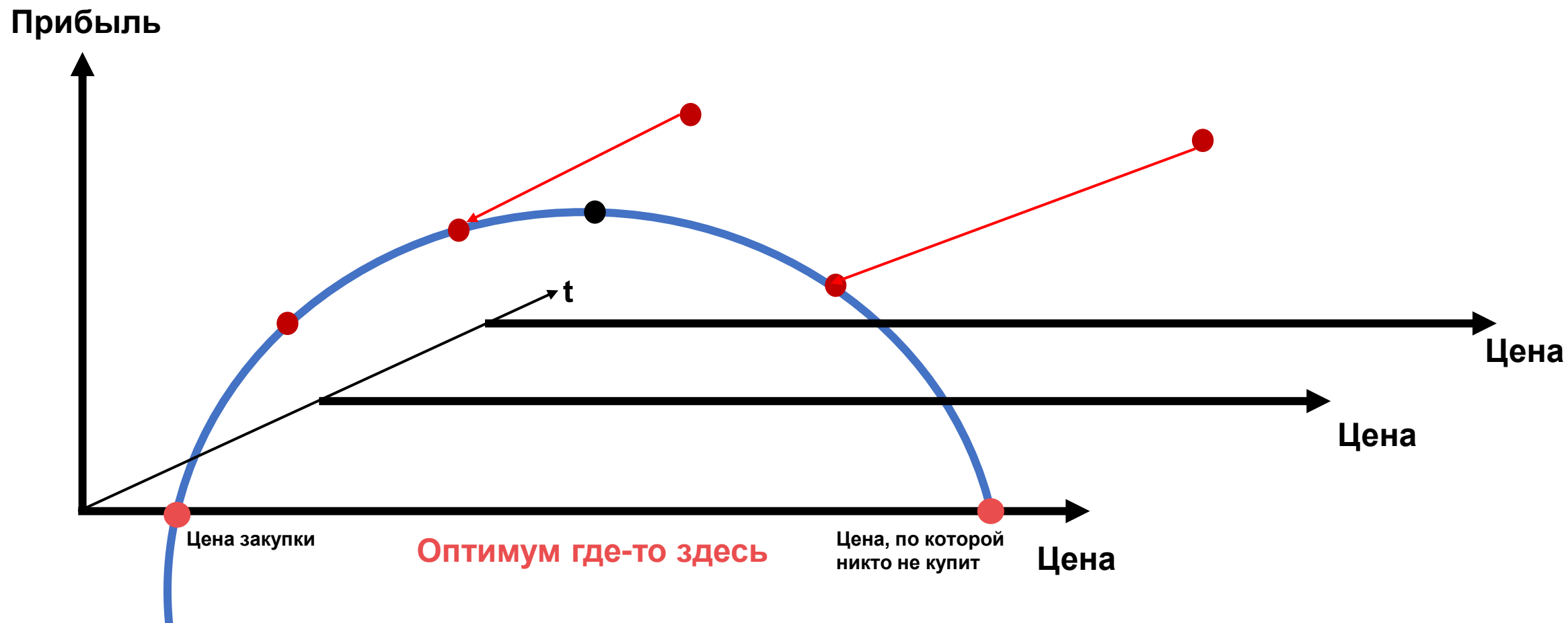
$$y(m, \xi) = \frac{Q(m)}{Q_t} m$$

Обратите внимание, что это не повлияет на поиск  $\operatorname{argmax}$ .

$$\operatorname{argmax}_m \frac{Q(m)}{Q_t} m = \operatorname{argmax}_m Q(m)m$$

Но удобнее обрабатывать выборку.

# Задача ценообразования в динамике





# Можно использовать «нормировку»

На самом деле ничего не мешает, рассмотреть и более общий случай.

То есть наши параметры не только зависят от  $t$ , но и от всех других факторов. Строить отдельно «макро» модель и нормировать на нее.

$$y(m, \xi) = \frac{Q(m)}{Q_x} m$$

# Можно использовать «нормировку»

«Макро» модель, подразумевает использование «понятных» (объяснимых) внешних факторов, которые в целом не зависят от  $m$  и влияют глобально на спрос:

- Погода
- Курс валют (одинаково, но на разные группы товаров по-разному)
- Биржевая цена

Но есть важные факторы, которые не обладают таким свойством. Например, «цена основного конкурента». Спрос ведет себя по-разному, если ваша цена дороже или дешевле этой цены.

# Можно использовать «нормировку»

Все это можно свести к самому, общему случаю, разумеется.

$$y(m, \xi) = Q(m, x)m$$

Здесь есть 2 варианта:

- Общая параметрическая модель спроса  $Q(m, x)$  - см. лекции №3.
- Считать, что параметры зависят от  $x$ . А характер параметризации остается – см. выше.

Пример 1.

$$Q(m, x) = \max(-am + b + Wxm, 0)$$

# Характер параметр. сохраняется

Есть реализации  $\xi_1, \xi_2, \xi_3 \dots$  Она порождает:

$$y_1(m, x) = Q_{x=x_1}(m)t \rightarrow a_1, b_1 \dots$$

$$y_2(m, x) = Q_{x=x_2}(m)t \rightarrow a_2, b_2 \dots$$

$$y_3(m, x) = Q_{x=x_3}(m)t \rightarrow a_3, b_3 \dots$$

Важно, что мы не агрегируем выборку, а считаем, что это независимые наблюдения по  $t$ .

- Для решения проблемы «старта». На первых нескольких итерациях считаем, что  $a, b$  не зависят от  $x$ . (статическая часть)
- Повторять в цикле (динамическая часть):
  - После этого решаем задачу регрессии  $(x, a, b)$ . Восстанавливаем  $a(x), b(x)$ .
  - При поиске оптимального  $m$  на следующем шаге
    - Спрогнозировать  $x_{n+1}$  на следующий шаг.
    - Подставляем  $a = a(x_{n+1})$   $b = b(x_{n+1})$  в функцию  $Q$ .

# Модель потребителя

# Модель потребителя



Общая идея: считаем, что есть множество потребителей (агентов). Каждый потребитель принимает решение о покупке (не покупке) товаров исходя из:

- Цен на товары
- Характеристик товаров
- Наблюдаемых внешних факторов
- ненаблюдаемых внешних факторов

Конечный спрос формирует поток таких агентов.

# Модель потребителя

Чуть более строго есть множество агентов  $A = \{a_1, a_2, a_3 \dots, a_N\}$

$$a_i = a_i(x, P, C, \eta) \rightarrow Q$$

$P$  – вектор цен на все товары

$C$  – матрица товаров, характеристик

$x$  – наблюдаемые внешние факторы

$\eta$  – шум (случайность при принятии решения)

Результатом функции является случайный вектор  $Q$  - количество покупок каждого товара.

# Модель потребителя

Совокупный спрос:

$$Q_{total} = \sum_i^N a_i \alpha_i ,$$

$\alpha_i$  - кол-во агентов  $i$ -ого типа.

Задача продавца найти оптимальный вектор цен  $P(x, C, \alpha)$ , как функция от факторов, характеристик и распределения агентов. Если  $a_i$  и  $\alpha_i$  известны – то это задача на стыке имитационного моделирования и оптимизации многомерной функции. Есть много разных техник для решения.



# Модель потребителя



А что делать если  $a_i$  неизвестны? Печалиться. ☺

В общем случае решить задачу, очевидно невозможно.

# Как устроены агенты?

Агентов удобно представлять в виде решающего дерева. Где на последних листьях находится кол-во товаров.

Если дополнительно предположить, что агенты «не сильно шумные», тогда можно сформулировать следующую задачу.

$$Q = f(x, p) = \sum_i^N a_i(x, p, \eta) \alpha_i$$
$$\hat{f}(x, p) = \sum_i^N \hat{a}_i(x, p) \hat{\alpha}_i,$$

Где  $\hat{a}_i(x, p)$  - решающие деревья.

# Как устроены агенты?



Сумма решающих деревьев? Где-то это уже было?

Random forest. Ну почти 😊.

# Модель потребителя



Если  $N$  – сравнительно небольшое, то и  $\alpha_i$  известны – то можно перестать печалиться. ☺

Вопросы:

Откуда можно узнать  $N$ ?

Как можно посчитать  $\alpha_i$ ?

# Модель потребителя



Если  $N$  – сравнительно небольшое, то и  $\alpha_i$  известны – то можно перестать печалиться. 😊

Вопросы:

Откуда можно узнать  $N$ ? Кластеризация клиентов

Как можно посчитать  $\alpha_i$ ? Кластеризация клиентов 😊

# Модель потребителя



Если кластеризация *достаточно хорошая* – то можно строить отдельно  $a_i(x, p, \eta)$ . В качестве таргета, используя продажи для каждой группы отдельно.

# Кластеризация!



	Батончик Snickers	Мороженое Baskin Robbins	Горький шоколад	Печенье Oreo
Пользователь1	10	9	1	7
Пользователь2	0	9	2	0
Пользователь3	1	0	6	0
Пользователь4	3	0	4	10
Пользователь5	0	1	0	0
Пользователь6	0	0	3	6

# Кластеризация пользователей



На самом деле мы уже умеем кластеризовать товары.

Можно делать все тоже самое с пользователями:

- Получить эмбединги
- Решить задачу кластеризации в полученном пространстве

Для эмбедингов можно использовать SVD, NNMF.



# Кластеризация пользователей



Кластеризация строится в динамике и анализируется «устойчивость» кластера.

Простая мера устойчивости:

- Сколько пользователей переходят из кластера в другие кластеры.
- Сколько приходят из других в данный.

0 – идеальная устойчивость.

# Как построить решающие функции



Для задачи построения модели потребления используют дополнительную кластеризацию, строят эмбединги по кластерам товаров.

- Сначала решается задача кластеризации товаров.
- Затем для каждого кластера строится отдельный эмбединг, и там решается отдельная кластеризация по пользователям

Дальше находятся «пограничные» пользователи (неустойчивые).

# Кластеризация!

	Кластер1. Товар 1.	Кластер1. Товар 2.	Кластер2. Товар 1.	Кластер2. Товар 2.
Пользователь1	10	9	1	7
Пользователь2	0	9	2	0
Пользователь3	1	0	6	0
Пользователь4	3	0	4	10
Пользователь5	0	1	0	0
Пользователь6	0	0	3	6

# Неустойчивые пользователи



Что это значит?

Они покупают в одном «классе», но иногда заменяют свои покупки другим «классом».

Пример.

Пользователь раз в 2 недели покупает рис и так продолжается 3 итерации, затем он не покупает рис, а покупает чечевицу, потом снова продолжает покупать рис.

# Неустойчивые пользователи

Неуверенность – это и есть «случайность» в принятиях решения пользователей (кластеров пользователей).

Таким образом, можно построить функцию принятия решения пользователей в каждом независимом кластере.

Если быть точным, ее конечное дискретное распределение.

Молоко 3,2% - 1 (не вероятность, а штуки)

Молоко 3,5% - 0.28 (не целое означает, что покупает не всегда)

Сливки – 0.01

...

# Неустойчивые пользователи

Эту функцию можно развернуть наверх, анализируя другие факторы.

В рамках нашей задачи нас разумеется больше всего интересует цена.

Пример.

Если **цена молока 3.5% <= цены на молоко 3.2%** тогда

0.5 молока 3.5%

иначе 0.5 молока 3.2%

...

# Неустойчивые пользователи

Мы хотим получить дерево решений.

$$a_i(P_k, \eta_1, \eta_2) \rightarrow Q$$

Обучающая выборка:  $(P_k, Q) \rightarrow a_i(P_k) = Q$

$$a_i(P_k) = [Q, Q_2(\eta_2)] \quad k = 1, n$$

Пример.

**Молоко 3,2% - 1 (не вероятность, а штуки)**

**Молоко 3,5% - 0.28 (не целое означает, что покупает не всегда)**

Сливки – 0.01 – рандом

$Q_2(\eta_2)$  - это весь вектор «пограничников».

# Что мы получили



$$Q(P) = \sum_i^N a_i(P) \alpha_i$$

Дальше можно оптимизировать по вектору  $P$ .



## 8. Непараметрические методы

# Формальная постановка задачи

Максимизировать прибыль  $y(p, \xi)$  по цене  $p \in [p, +\infty)$  .

4 разные постановки:

1.  $p^* = \operatorname{argmax}_p E[y(p_n, \xi)]$  за  $n$  итераций.
2.  $p^* = \operatorname{argmax}_p E[y(p, \xi)]$  за минимальное время.
3.  $\max_p \sum E[y(p, \xi)]$  Оптимизация траектории.
4. Найти зависимость  $y(p)$

$E[\cdot]$  - может быть  
любым оператором.  
Можно считать, как  
мат. ожидание.



Основная проблема  
параметризации – где взять  
«хорошую» параметризацию?

# Формальная постановка задачи

Максимизировать прибыль  $y(p, \xi)$  по цене  $p \in [p, +\infty)$  .

4 разные постановки:

1.  $p^* = \operatorname{argmax}_p E[y(p_n, \xi)]$  за  $n$  итераций.
2.  $p^* = \operatorname{argmax}_p E[y(p, \xi)]$  за минимальное время.
3.  $\max_p \sum E[y(p, \xi)]$  Оптимизация траектории.
4. Найти зависимость  $y(p)$

$E[\cdot]$  - может быть  
любым оператором.  
Можно считать, как  
мат. ожидание.

# Многорукий бандит

**Многорукий бандит (в теории вероятности)** — это проблема, в которой фиксированный ограниченный набор ресурсов должен быть распределен между конкурирующими (альтернативными) вариантами так, чтобы это максимизировало ожидаемую выгоду. Свойства каждого выбора («ручки») известны только по наблюдению.



# Многорукый бандит

## Постановка задачи:

- На каждом шаге можно выбрать некоторую «ручку» (цену)  $x_i \in X$ .
- Окружающая среда сообщает награду  $y$  (прибыль), которая получается в результате этого действия.
- **Цель:** выбрать оптимальную стратегию выбора «ручек», т.е. последовательность  $x = (x_1, x_2, \dots, x_N)$ , чтобы максимизировать:

$$\operatorname{argmax}_x \sum E[y(x, \xi, t)]$$

# Многорукий бандит



## Алгоритмы:

- Жадные (greedy) стратегии
- Upper Confidence Bound (UCB)
- Томпсоновское семплирование

# Жадные стратегии (greedy)

## Основная идея:

- Выбираем ручку (цену), которая в среднем приносит наибольший выигрыш (доход)
- Проверяем другие ручки (цены) с вероятностью  $\epsilon$
- $\epsilon$  - может варьироваться или быть постоянной.



# Upper Confidence Bound (UCB)

## Основная идея:

- Выбираем ручку в соответствии с формулой:

$$\arg \max_{x_i} \left( E[\hat{y}(x_i)] + \sqrt{\frac{2 \ln n}{n_i}} \right),$$

где  $n$  – сколько всего раз мы дергали все «ручки», а  $n_i$  – сколько раз мы дергали  $i$ -ую ручку.

$\hat{y}(*)$  – оценка функции выигрыша (в общем случае случайная величина).

$\hat{y}(x_i)$  – значение выигрыша в точке  $x_i$

# Томпсоновское семплирование



## Основная идея:

- Это байесовский подход!
- Для каждой ручки строится сопряженное распределение.
- Семплируем из распределения каждой ручки.
- Выбираем максимум.
- Получаем результаты – обновляем параметры сопряженных распределений.

# На практике



1. Бандиты хороши с точки зрения математики.
2. Томпосновское семплирование дает лучший результат.
3. UCSB проигрывает, но не сильно.
4. Жадные стратегии самые неэффективные.



Одномерный случай

# Добавим специфику

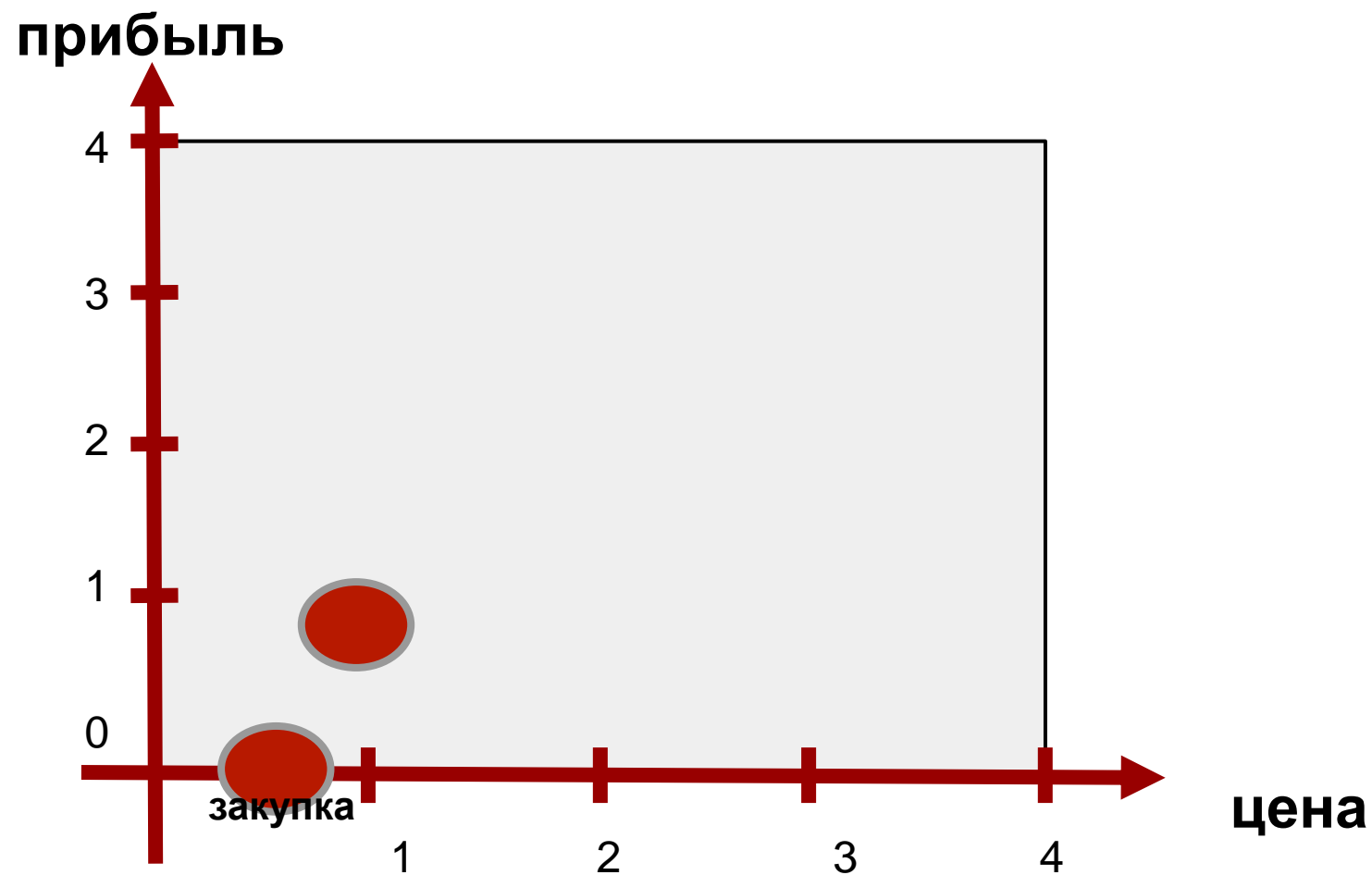
Нашу целевую функцию (прибыль) можно записать в виде:

$$y(x) = Q(x)(x - \underline{x}),$$

где  $Q(x)$  - это кол-во проданных товаров по цене («спрос»).

1. Есть история продаж (часто это всего одна цена).
2. О функции  $y(x)$  известно:
  - Известна цена закупки  $\underline{x}$ , т.е. там, где будет точно 0.
  - Максимум функции  $y(x^*)$  находится в точке  $\underline{x} < x^*$ .
  - $Q(x)$  - неотрицательная и неубывающая «в среднем» по  $x$ .
3. Если известны цены конкурентов, то на этапе тестирования, можно также искать максимум в  $x^* < \bar{x}$ . Где  $\bar{x}$  - максимальная цена на рынке (или экспертная оценка).

# Задача ценообразования



# Параметрические подходы

Нашу целевую функцию можно записать в виде:

$$r(x) = Q(x)(x - \underline{x}),$$

где  $Q(x)$  - это кол-во проданных товаров по цене («спрос»).

Линейная функция:  $Q(x) = \max(-ax + b, 0)$

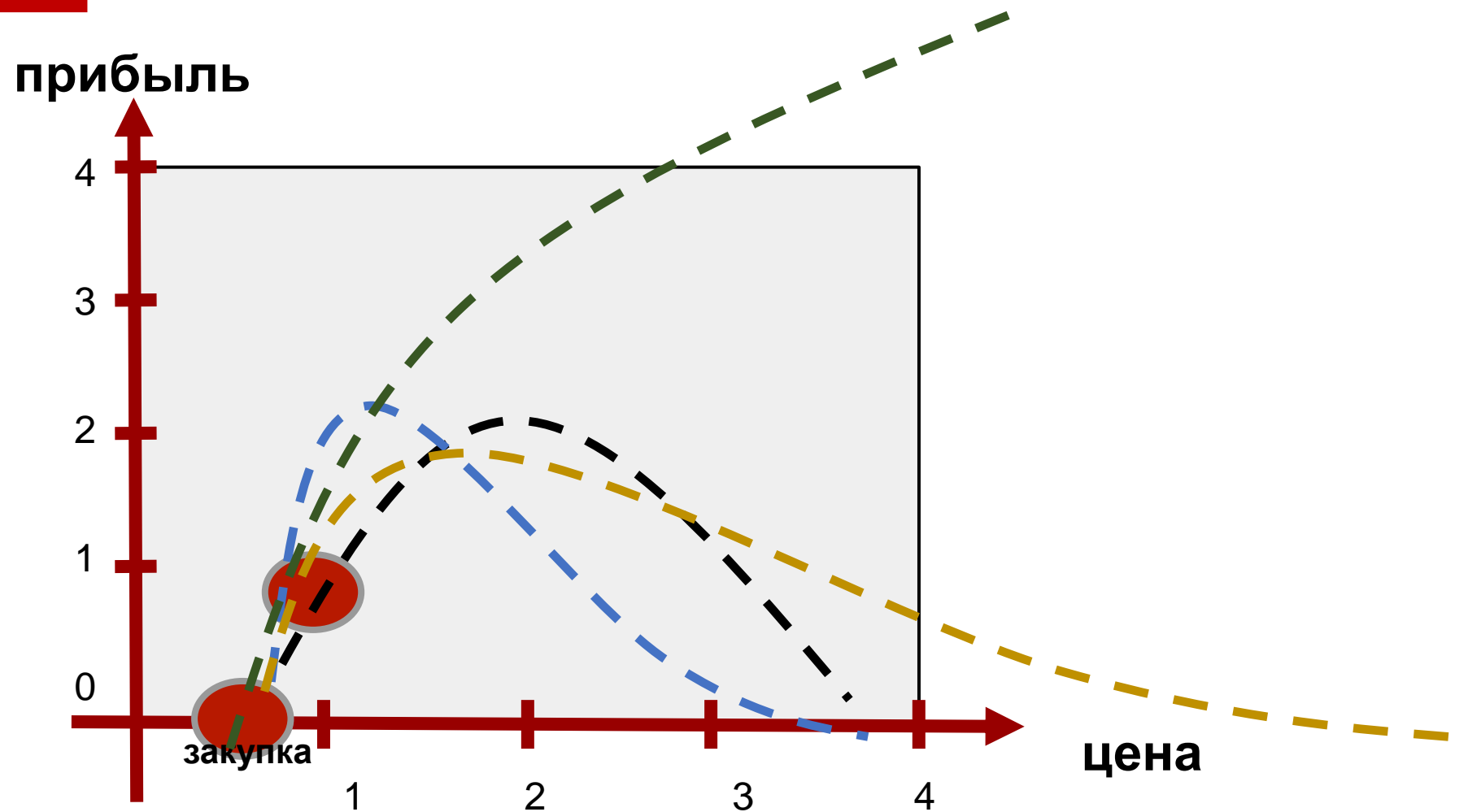
Гиперболическая функция:  $Q(x) = \max(-a/x + b, 0)$

Экспоненциальная функция:  $Q(x) = \max(-\exp(ax + b) + c, 0)$

Показательная функция:  $Q(x) = \max(ax^b + c, 0)$

И т.д.

# Задача ценообразования





# Изменим пространство $x$

1. Неудобно работать с бесконечным числом ручек.
2. Нужно учесть специфику того, что чем меньше цена, тем больше влияет ее абсолютное изменение. Ширина бина должна расти с ростом цен.
3. При сокращении числа ручек, можно каждую из оставшихся заново дискретизовать.

Цена от	Цена до	Ширина бина	Средняя
1000	1041	41	1020
1041	1084	43	1062
1084	1129	45	1106
1129	1177	48	1153
1177	1229	52	1203
1229	1285	56	1257
1285	1346	61	1315
1346	1413	67	1379
1413	1487	74	1450
1487	1570	83	1528
1570	1663	93	1616
1663	1769	106	1716
1769	1891	122	1830
1891	2034	143	1962
2034	2203	169	2118

# Давайте еще раз посмотрим UCSB

$$\arg \max_{x_i} \left( E[\hat{y}(x_i)] + \sqrt{\frac{2 \ln n}{n_i}} \right)$$

Теперь  $\hat{y}(x_i)$  - это оценка нашей параметризованной функции в точке  $x_i$ , а  $x_i$  («ручки») - это бины.

На каждом шаге:


- Пересчитываем параметры  $\hat{y}(\cdot)$ . Строим оценки для бинов  $\hat{y}(x_i)$ .
- Считаем статистику бинов. Когда какие-то бины начинают появляться часто (больше 33%) – проводим дискретизацию в них. Считая в начале, что статистика для частей бина такая же как «родительского» бина. [Здесь есть более продвинутые результаты]

# А как выбрать функцию $Q(x)$ ?

В общем случае, очевидно нельзя выбрать сразу «правильную» параметрическую функцию.

На самом деле, нам неважно, насколько она хороша.

**Важно:** Чтобы она верно указывала на оптимум!



А давайте еще добавим  
active learning!

# Несогласие в комитете (Query By Committee)

Идея Query By Committee(QBC): у нас есть  $J$  алгоритмов (параметрических моделей). Они образуют *комитет*. Выбираем  $x_i$  – где модели между собой максимально расходятся, чтобы снизить неопределенность в будущем. Так мы сможем быстрее найти лучший алгоритм.

Что значит максимально расходятся?

- 1) Максимальная выборочная дисперсия
- 2) Максимальный (интерквантильный) размах
- 3) Взвешенная оценка (1) и (2)
- 4) ...

# UCB + QBC

Можно выбирать исходя из следующих соображений:

$$\arg \max_{x_i} \left( \overset{\text{UCB}}{\lambda \left( \frac{1}{JA} \sum_{j=1}^J E[\hat{y}_j(x_i) \alpha[y_j(x_i)]] + \sqrt{\frac{2 \ln n}{n_i}} \right)} + (1 - \lambda) \overset{\text{QBC}}{\left( \frac{1}{J} \sqrt{\sum_{j=1}^J D[\hat{y}_j(x_i)]} \right)} \right),$$

где  $\lambda \in (0,1)$  – это некоторый параметр, с которым мы учитываем «вес» оценки в точке (UCB), а с  $1 - \lambda$  учитываем вес «расхождения» в комитете.

$\frac{1}{J} \sum_{j=1}^J E[\hat{y}_j(x_i) \alpha[y_j(x_i)]]$  — среднее по всем алгоритмам. Где  $\alpha[y_j(x_i)]$  – «вес» (связанный с оценкой качества)  $j$ -ого алгоритма в точке  $x_i$ .  $A = \sum_{j=1}^J \alpha[y_j]$  — нормировочная константа.

$\frac{1}{J} \sqrt{\sum_{j=1}^J D[\hat{y}_j(x_i)]}$  — «расхождение» в комитете. Можно использовать другие меры расхождения.

# Оценка качества алгоритма

Функция  $\alpha[y_j(x_i)]$  решает сразу две проблемы:

- Дает приоритет более качественным алгоритмам.
- Учитывает риск итераций в точках, удаленных от известных (уже тестированных)  $x_k$  точек. (Очень важная эвристика!)

# Оценка качества алгоритма

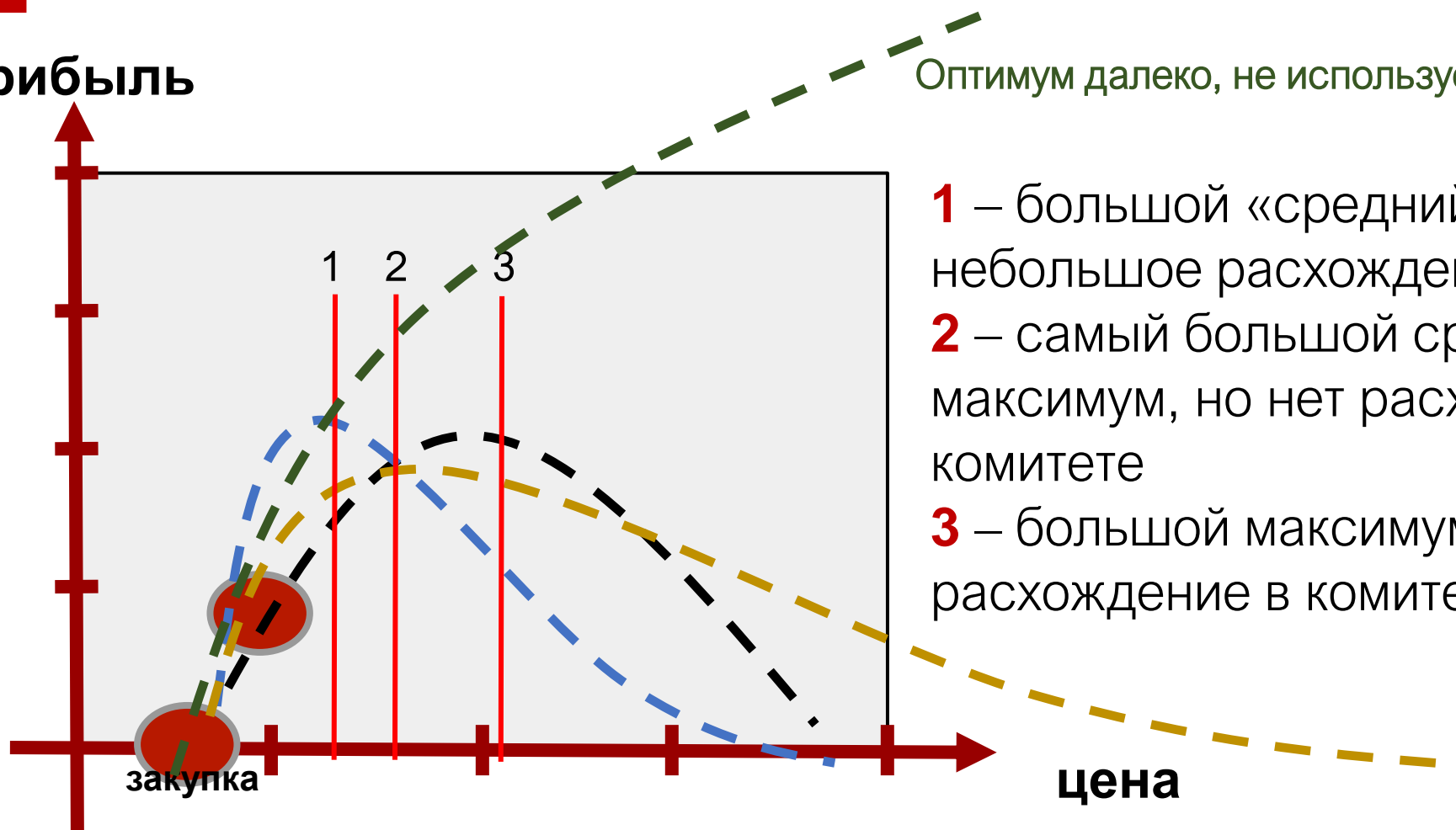
Возможные варианты учета удаления от известных точек:

- Ширина доверительного интервала в точке  $x_i$ . (квантильные регрессии)
- Можно по одному удалять уже известные прецеденты.  
Строить  $\widetilde{y}_j^k(x_i)$  для  $k$ -го удаленного элемента. Использовать  $\max_k \widetilde{y}_j^k(x_i) - \min_k \widetilde{y}_j^k(x_i)$ , как меру риска.
- Можно отдельно штрафовать за отдаление по аргументу  $\min_k |x_i - x_k|$ . Например, линейно или квадратично.
- Сильно уменьшать вес алгоритма (до 0), если его оптимум находится за  $\bar{x}$ .

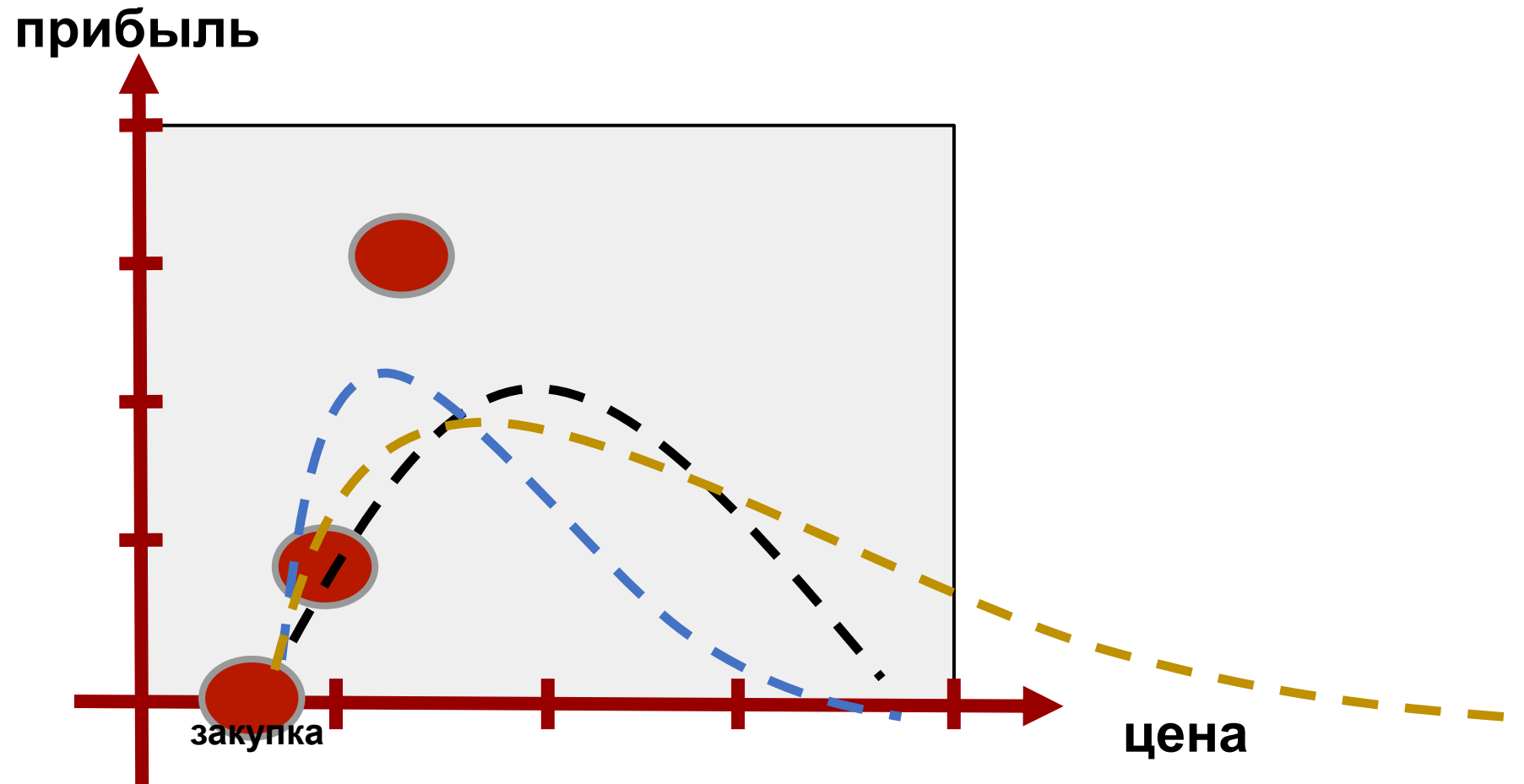


# Алгоритм. Выбираем точку

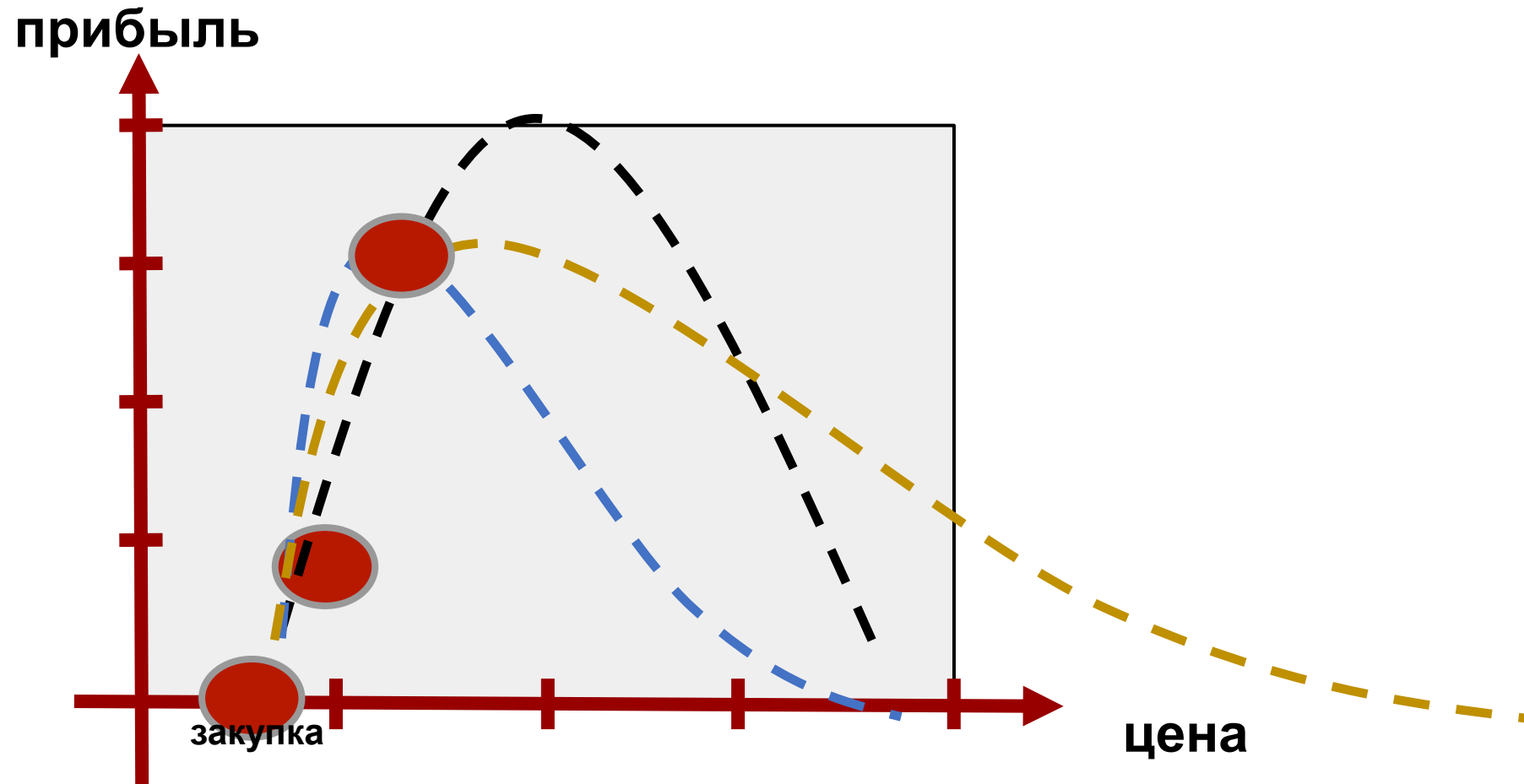
прибыль



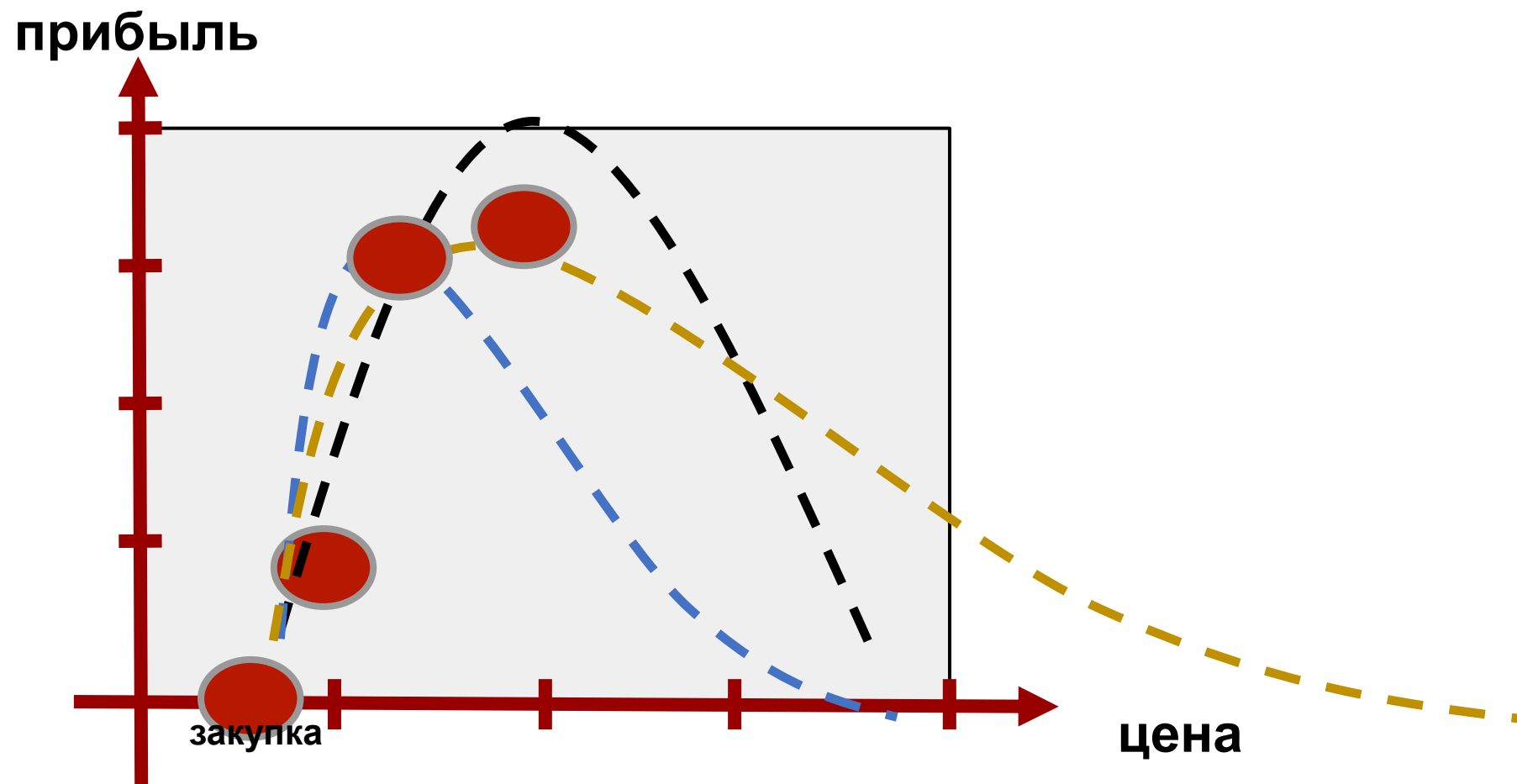
# Алгоритм. Получаем оценку



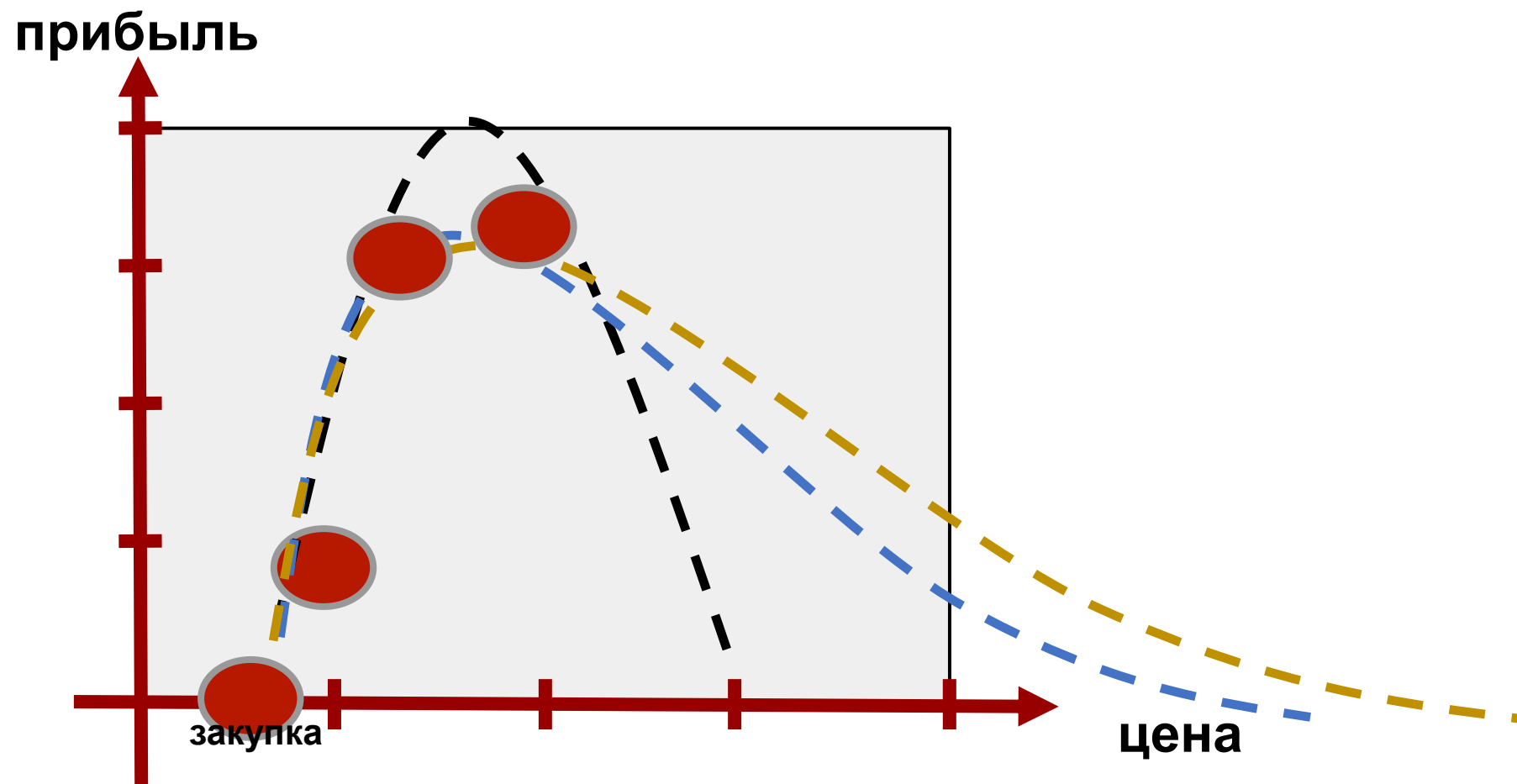
# Алгоритм. Пересчитываем параметры



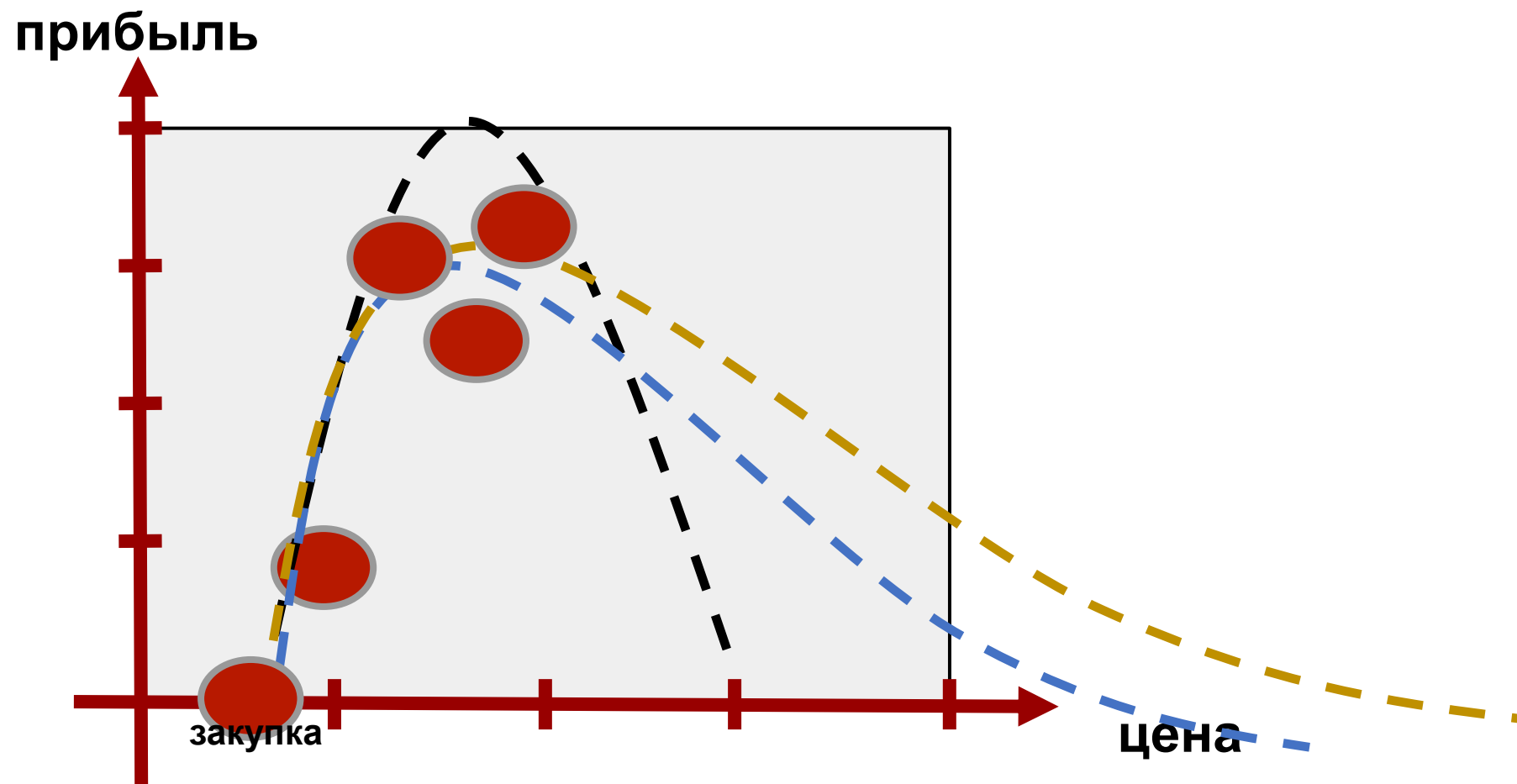
# Алгоритм. Выбираем точку



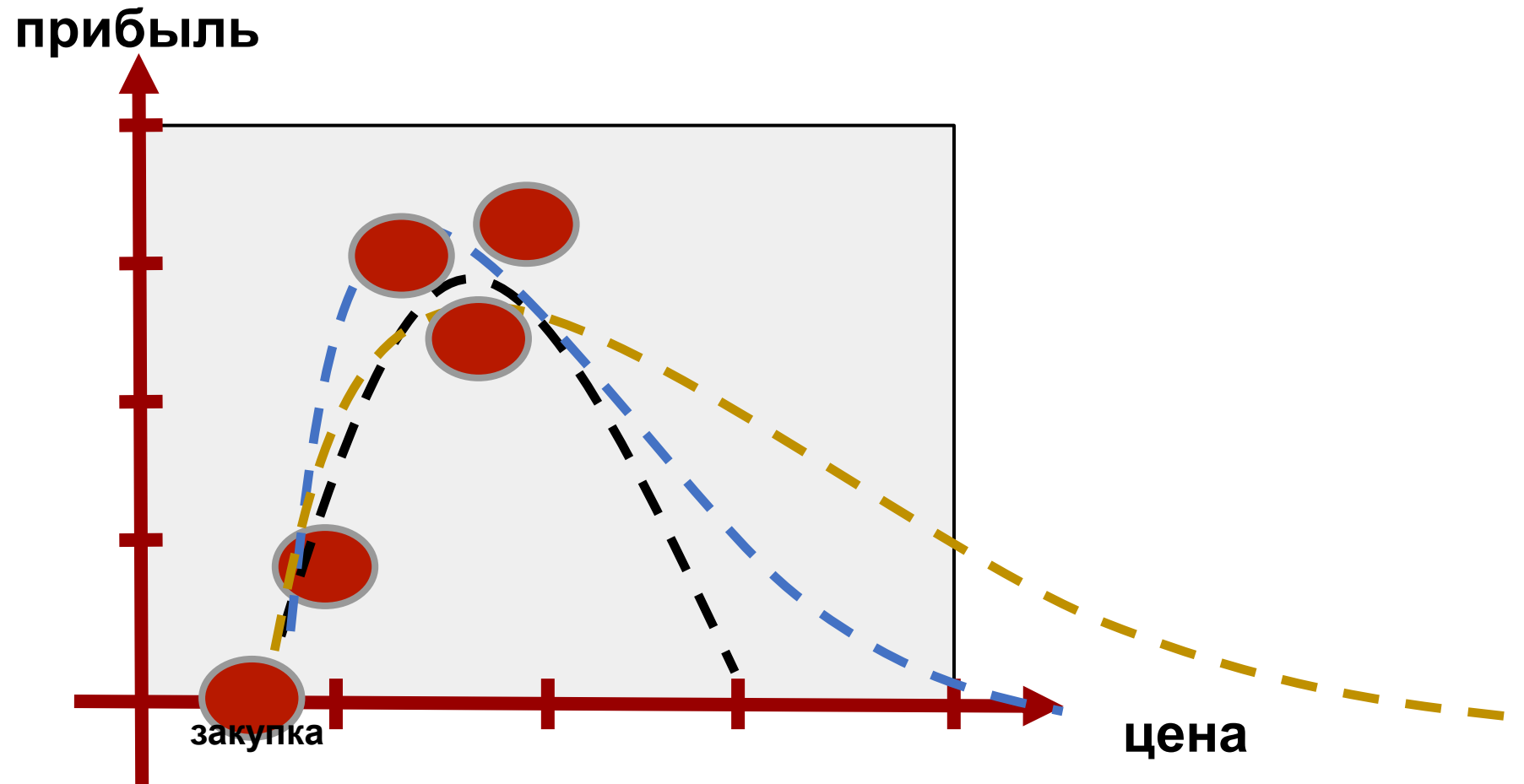
# Алгоритм. Пересчитываем параметры




# Алгоритм. Выбираем точку



# Алгоритм. Пересчитываем параметры





Как обобщить на многомерный  
случай?