Конспект по теме «Юнит-экономика»

Экономика одной продажи

Расходы предприятия делят на переменные и постоянные.

Переменные расходы связаны с объёмом продаж. Например, расходы на закупку товаров. Чтобы продать 20 велосипедов, нужно сначала купить 20 велосипедов.

Постоянные расходы не меняются вместе с выручкой. Сколько бы ни было продаж, за аренду офиса нужно платить каждый месяц.

Подход к анализу бизнеса, при котором рассчитывают количество продаж, которого хватит, чтобы выйти в плюс, называют экономикой одной продажи, или юнит-экономикой.

Расчёт экономики одной продажи в Python

Чтобы рассчитать экономику одной продажи:

1. Посчитайте рентабельность бизнеса на одной продаже. Принимайте во внимание только переменные расходы.
2. Если рентабельность отрицательная, значит, экономика не сходится. Нужно снижать расходы, либо повышать продажи. В противном случае бизнес придётся закрыть.
3. Если с учётом переменных расходов рентабельность положительная, прибавьте постоянные расходы. Выясните, какой объём продаж должен быть, чтобы бизнес приносил прибыль.

Для подсчёта юнит-экономики для заданного бюджета подойдёт такая функция:

```
def unit_economics(marketing):
    items_sold = marketing / one_unit_var_costs['marketing']
    revenue = one_unit_revenue  * items_sold
    var_costs = one_unit_var_costs * items_sold
    return revenue - sum(var_costs) - sum(fixed_costs)
```

Экономика одного покупателя: LTV, CAC и ROI

Экономика одной продажи не учитывает повторные покупки, а они важны. Чтобы включить в анализ повторные покупки, считают экономику одного покупателя. Три самые важные метрики в этом методе — LTV, CAC и ROI.

LTV — общая сумма денег, которую один клиент в среднем приносит компании со всех своих покупок. Чаще всего анализируют LTV за первые 1, 3, 7 и 14 дней после регистрации. `LTV = Общая выручка на текущий день / Размер когорты`.

CAC — стоимость привлечения одного клиента. Сумма денег, в которую компании обходится каждый новый клиент. В сущности, CAC — это инвестиции в маркетинг. `CAC = Расходы на рекламу / Размер когорты`.

ROI — окупаемость инвестиций. В экономике одного покупателя эта метрика показывает, на сколько процентов LTV превысил CAC, то есть на сколько процентов «окупились» клиенты. `ROI = LTV / CAC`.

LTV, ARPU и ARPPU. Методы оплаты рекламы

Иногда вместо LTV используют показатели ARPU или ARPPU.

ARPU — это то же самое, что LTV.

ARPPU — LTV, рассчитанный с учётом только платящих пользователей когорты. Выручку с накоплением делят на число пользователей, совершивших хотя бы одну покупку.

Выбор между ARPU и ARPPU отчасти зависит от способа оплаты рекламы, которым пользуется компания. Самые распространённые — CPM, CPC, CPL/CPI и CPA.

CPM, или **Cost Per Mille**, — оплата за тысячу просмотров рекламных объявлений. Самая «обезличенная» модель: не позволяет идентифицировать пользователей.

CPC, или **Cost Per Click**, — оплата за каждого пользователя, кликнувшего на рекламное объявление, то есть перешедшего на сайт. Для идентификации можно использовать куки.

CPL, или **Cost Per Lead**, — оплата за каждого пользователя, оставившего свои контакты. Аналог CPL в мобильных приложениях — оплата за установку, CPI, или Cost Per Install. Такая модель позволяет однозначно идентифицировать привлечённых пользователей по логину, почте или номеру телефона или рекламному идентификатору устройства.

CPA, или **Cost Per Action**, — оплата за определённое действие, совершённое пользователем. Например, за покупку. Самая дорогая, но и самая информативная модель.

Расчёт LTV в Python

Чтобы рассчитать LTV в Python, выполняют такие шаги:

- Получить пользовательские профили и данные о покупках.
- Добавить данные о покупках в профили.
- Рассчитать лайфтайм пользователя для каждой покупки.
- Построить таблицу выручки. То есть сводную таблицу, в которой названия строк — это даты первого посещения пользователей, названия столбцов — лайфтайм, а значения в «ячейках» — выручка.
- Посчитать сумму с накоплением для каждой строки таблицы выручки.
- Вычислить размеры когорт и занести результаты в отдельную таблицу.
- Объединить таблицы размеров когорт и выручки.
- Посчитать LTV: разделить каждую «ячейку» таблицы выручки на соответствующий размер когорты.

Расчёт CAC и ROI в Python

Чтобы рассчитать CAC в Python, выполняют такие шаги:

- Передать функции для создания профилей данные о тратах на рекламу.
- Объединить данные о тратах на рекламу и новых пользователях.
- Вычислить CAC: разделить рекламные расходы на количество новых пользователей.

- Добавить CAC для каждой даты привлечения и источника в профили.

Разделив LTV на CAC, получаем ROI — возврат на инвестиции. При расчёте CAC важно учитывать момент и горизонт анализа, используемые при расчёте LTV.

Как проверить себя. Удержание, конверсия, LTV, CAC и ROI

Разберём некоторые особенности метрик, которые позволят устранить сомнения в корректности расчётов или вовремя заподозрить неладное.

Признаки верного расчёта удержания

- Сумма размеров когорт равна числу новых клиентов в изучаемый период.
- Сумма размеров платящих когорт равна числу покупателей в изучаемый период.
- Удержание убывает по экспоненциальному закону.
- Удержание неплатящих убывает быстрее, чем удержание платящих.

Признаки верного расчёта конверсии

- Сумма размеров когорт равна общему числу новых клиентов в изучаемый период.
- Кривая конверсии плавно растёт от нуля в направлении единицы.
- Кривая конверсии не снижается.
- В таблице конверсии нет значений, превышающих единицу.
- Количество новых покупателей равно числу новых клиентов, умноженному на общую конверсию.

Признаки верного расчёта LTV

- Сумма размеров когорт равна общему числу новых клиентов в изучаемый период.
- Кривая LTV плавно растёт от нуля с возможным пересечением единицы.
- Кривая LTV не снижается.

- Общая стоимость покупок новых клиентов равна максимальному LTV, умноженному на число новых клиентов.

Признаки верного расчёта CAC

- CAC из таблицы ROI, умноженный на размер когорты, равен сумме рекламных трат за изучаемый период.

Признаки верного расчёта ROI

- Реалистичный ROI находится в пределах от 0 до 3.

Продвинутая визуализация. Параметр sharey

Параметр `sharey` функции `subplot()` задаёт расположение графика, вертикальную ось которого следует перенять.

```
# график во второй ячейке таблицы графиков делит
# вертикальную ось с графиком в первой ячейке

plt.subplot(1, 2, 2, sharey=plt.subplot(1, 2, 1))
```

Продвинутая визуализация. Скользящее среднее

Метод скользящего среднего применяют, когда хотят уменьшить шумы и сгладить график, выделив общую тенденцию изменения величины. В исходной последовательности поочерёдно составляют группы значений, количество которых равно ширине окна сглаживания, и считают в них среднее. Получается новая, более короткая последовательность, по которой строят «сглаженный» график.

Метод `rolling()` задаёт окно, которое «едет» по Series и в котором можно применять математические функции. Чтобы получить скользящее среднее, применяют `mean()`.

```
# скользящее среднее с шириной окна 4

test = [1, 2, 3, 4, 5, 6, 7, 8]
pd.Series(test).rolling(4).mean()
```