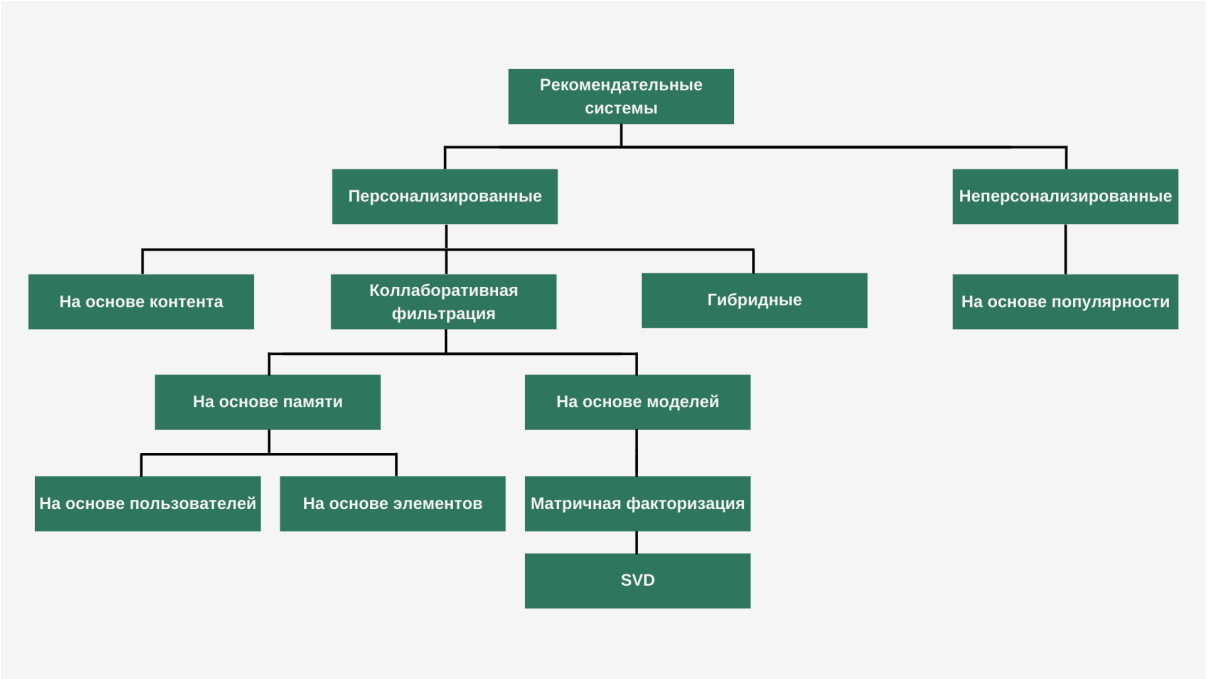
Подходы к построению рекомендательных систем

Методы построения рекомендательных систем можно представить следующим образом:



Данные для рекомендательной системы

Явный сбор данных (Explicit Feedback)

В рамках явного сбора данных получают ту информацию, которую передают сами пользователи. Явные отзывы учитывают мнение пользователя о том, насколько ему понравился или не понравился продукт.

Использование таких данных сопряжено с рядом **проблем**:

- ➔ После сбора явных отзывов вы с большой вероятностью получите набор оценок с **ярко выраженной полярностью**.
- ➔ Явные отзывы **не учитывают контекст**, в котором был оценён тот или иной продукт.

- Также есть **сложности с выстраиванием системы оценок**: необходимо придумать такую шкалу, чтобы пользователям было легко выражать своё мнение и это действие не вызывало у них раздражение.

Неявный сбор данных (Implicit Feedback)

Примерами неявных данных являются история просмотров, клики по ссылкам, подсчёт количества проигрываний песни, процент прокрутки веб-страницы или даже движение курсора по странице.

Особенности неявных данных:

- Отсутствие прямого измерения негативных предпочтений.
- Возможность делать верные выводы по численному выражению обратной связи.
- Большой объём зашумлённых данных.

Метрики в рекомендательных системах

Все метрики для оценки качества РС можно разделить на две группы:

- **офлайн-метрики** (оценивают качество алгоритма);
- **онлайн-метрики** (оценивают производительность и бизнес-показатели).

Офлайн-метрики

Существует несколько категорий офлайн-метрик:

- **Prediction Accuracy** — оценка точности предсказываемого рейтинга.
- **Decision Support** — оценка релевантности рекомендаций.
- **Rank Accuracy** — оценка качества рекомендаций с учётом ранжирования.

Prediction Accuracy

Название	Формула	Описание
MAE (Mean Absolute Error)	$E(\|P-R\|)$	Среднее абсолютное отклонение
MSE (Mean Squared Error)	$E(\|P-R\|^2)$	Среднеквадратичная ошибка
RMSE (Root Mean Squared Error)	$\sqrt{E(\|P-R\|^2)}$	Корень из среднеквадратичной ошибки

Здесь P — предсказанные оценки, R — реально выставленные оценки, E — математическое ожидание.

Для того чтобы вычислить *MAE* и *RMSE* при построении рекомендательной системы, можно воспользоваться уже известными вам готовыми функциями из модуля *sklearn*:

- sklearn.metrics.mean_absolute_error(y_true, y_pred);
- sklearn.metrics.mean_squared_error(y_true, y_pred).

Decision Support

Точность рекомендательной системы ($Precision$): $P = \frac{\text{количество релевантных рекомендаций}}{\text{общее количество рекомендованных элементов}}$

Полнота рекомендательной системы ($Recall$): $R = \frac{\text{количество релевантных рекомендаций}}{\text{общее количество релевантных элементов}}$

Precision и recall в точке отсечения k, $P@k$ и $R@k$ — это просто *precision* и *recall*, рассчитанные с учётом только подмножества рекомендаций от ранга 1 до k.

Для оценки качества рекомендуем использовать модуль cute_ranking.

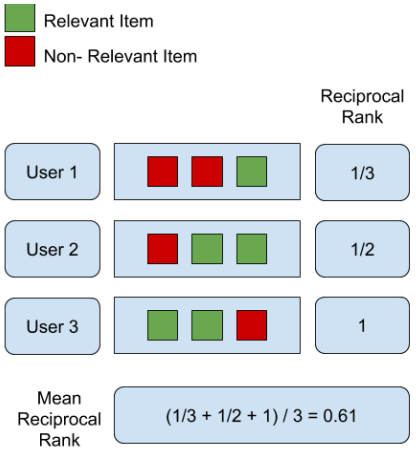
Rank Accuracy

MRR (средний реципрокный ранг)

$$\text{MRR} = \frac{1}{|Q|}\sum_{i=1}^{|Q|}\frac{1}{\text{rank}_i}$$

Здесь:
$|Q|$ — общее количество запросов;
$rank_i$ — позиция первого релевантного элемента для i-го запроса.

Ниже представлен пример вычисления *MRR*. Для каждого пользователя мы находим первый релевантный элемент и вычисляем, сколько элементов к этому моменту уже было предложено пользователю. Далее делим 1 на это количество. После этого находим среднее арифметическое для всех полученных долей.



Преимущества MRR

- Метод прост в вычислениях и легко интерпретируется.
- Метод уделяет большее внимание первому релевантному элементу списка, что в целом отражает логику рекомендательных систем.

Недостатки MRR

- Метрика фокусируется на одном элементе из списка и не оценивает остальные рекомендуемые элементы.
- Список с одним релевантным элементом имеет такой же вес в вычислении итогового показателя, как и список с большим количеством релевантных элементов. Это не всегда хорошо.
- Метрика плохо подходит для случаев, когда важно получить именно ряд рекомендаций, а не одну рекомендацию.

Для вычисления *MRR* в *Python* рекомендуем использовать функцию `mean_reciprocal_rank()` из библиотеки cute_ranking.

MAP (средняя точность)

Преимущества MAP

- Метрика естественным образом обрабатывает ранжирование списков рекомендованных элементов.
- Метрика способна придавать вес ошибкам пропорционально их месту в списке: больший вес — ошибкам в верхней части, меньший вес — ошибкам ниже по списку. Это соответствует необходимости показать как можно больше релевантных элементов в верхней части списка рекомендаций.

Недостатки MAP

- Метрика отлично подходит для бинарных (релевантных/нерелевантных) оценок, однако не подходит для рейтинговых числовых оценок.

NDCG (нормализованный дисконтированный кумулятивный выигрыш)

Представим, что поисковая система выдаёт пять статей с именами *D1*, *D2*, *D3*, *D4*, *D5*, которые выводятся в таком же порядке. Определим шкалу релевантности (0–3), где:

- 0 — не релевантно;
- 1–2 — в некоторой степени релевантно;
- 3 — полностью релевантно.

Предположим, статьи имеют следующие оценки релевантности:

D1 — 3;
D2 — 2;
D3 — 0;
D4 — 0;
D5 — 1.

Кумулятивный выигрыш представляет собой сумму этих оценок релевантности и может быть рассчитан как:

$$CG = \sum_{i=1}^{5} (rel)_i = 3 + 2 + 0 + 0 + 1 = 6$$

Здесь rel — оценка релевантности документа.

Дисконтированный кумулятивный выигрыш можно рассчитать по формуле:

$$DCG = \sum_{i=1}^{5} \frac{rel_i}{\log_2(i+1)}$$

Таким образом, дисконтированный кумулятивный выигрыш в приведённом выше примере составляет:

$$DCG_5 = \frac{3}{log_2(2)} + \frac{2}{log_2(3)} + \frac{0}{log_2(4)} + \frac{0}{log_2(5)} + \frac{1}{log_2(6)}$$
$$DCG_5 = 3 + \frac{2}{1.585} + 0 + 0 + \frac{1}{2.585}$$
$$DCG_5 = 3 + 1.26 + 0.3868$$
$$DCG_5 \simeq 4.67$$

Теперь нам нужно расположить статьи в порядке убывания рейтинга и рассчитать *DCG*, чтобы получить рейтинг идеального дисконтированного кумулятивного выигрыша *(IDCG)*:

$$IDCG_5 = \frac{3}{log_2(2)} + \frac{2}{log_2(3)} + \frac{1}{log_2(4)} + \frac{0}{log_2(5)} + \frac{0}{log_2(6)}$$
$$IDCG_5 = 3 + \frac{2}{1.585} + \frac{1}{2} + 0 + 0$$
$$IDCG_5 = 3 + 1.26 + 0.5$$
$$IDCG_5 = 4.76$$

Рассчитаем нормализованный *DCG* по следующей формуле:

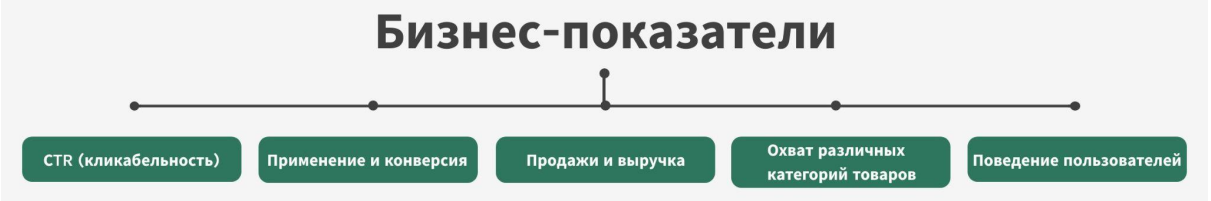
$$nDCG = \frac{DCG_5}{IDCG_5}$$
$$nDCG = \frac{4.67}{4.76}$$
$$nDCG \simeq 0.98$$

Можно получить то же значение, если воспользоваться готовой функцией из модуля *sklearn*:

```
from sklearn.metrics import ndcg_score, dcg_score
import numpy as np

true = np.asarray([[3, 2, 1, 0, 0]])
relevance = np.asarray([[3, 2, 0, 0, 1]])

print(ndcg_score(true, relevance))

#0.980840401274087
```

Онлайн-метрики

Выделяют пять групп **бизнес-показателей**, на которые рекомендательные системы оказывают самое существенное влияние:



Проблема холодного старта и popularity-based model

Можно выделить два вида проблем холодного старта:

- проблема холодного старта пользователя;
- проблема холодного старта продукта.

Холодный старт пользователя

Когда система сталкивается с новыми посетителями веб-сайта, не имеющими истории просмотров или известных предпочтений, создание персонализированного опыта для них становится сложной задачей, поскольку данные, обычно используемые для создания рекомендаций, отсутствуют.

Холодный старт продукта

Когда новый товар добавляется в интернет-магазин или когда свежий контент загружается на медиаплатформу, первое время о нём никто не знает. С нулевым количеством взаимодействий или оценок он практически невидим для рекомендательной системы независимо от того, насколько релевантным он будет для пользователей.

Popularity-based Model

Если к нам приходит клиент, про которого мы ничего не знаем, мы можем создать для него рекомендации с использованием *popularity-based*-модели. Это тип РС, которая формирует рекомендации на основе популярности продуктов.

Достоинства рекомендательной системы на основе популярности:

- Не страдает от проблем холодного старта.
- Нет необходимости в исторических данных для пользователя.

Недостатки рекомендательной системы на основе популярности:

- Не персонализирована.
- Система будет рекомендовать одинаковые продукты всем пользователям.