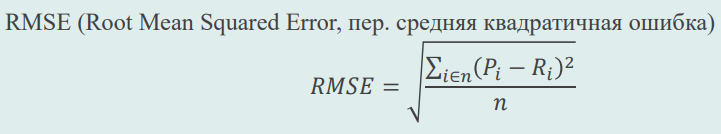
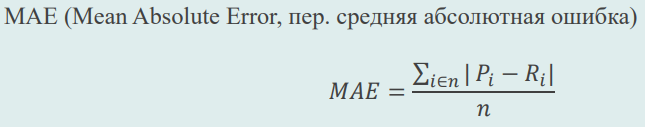
Какие показатели используются для оценки рекомендательных систем?

Оффлайн эксперимент: разделяем выборку на тестовую и обучающую, а затем оцениваем алгоритм с помощью RMSE, MAE.

Также можно провести онлайн-тестирование (A/B). Для некоторой части пользователей будет применён новый. А для другой группы прежний.

Метрики точности прогнозов (MAE, RMSE).





В каких случаях используют MSE, а в каких MAE?

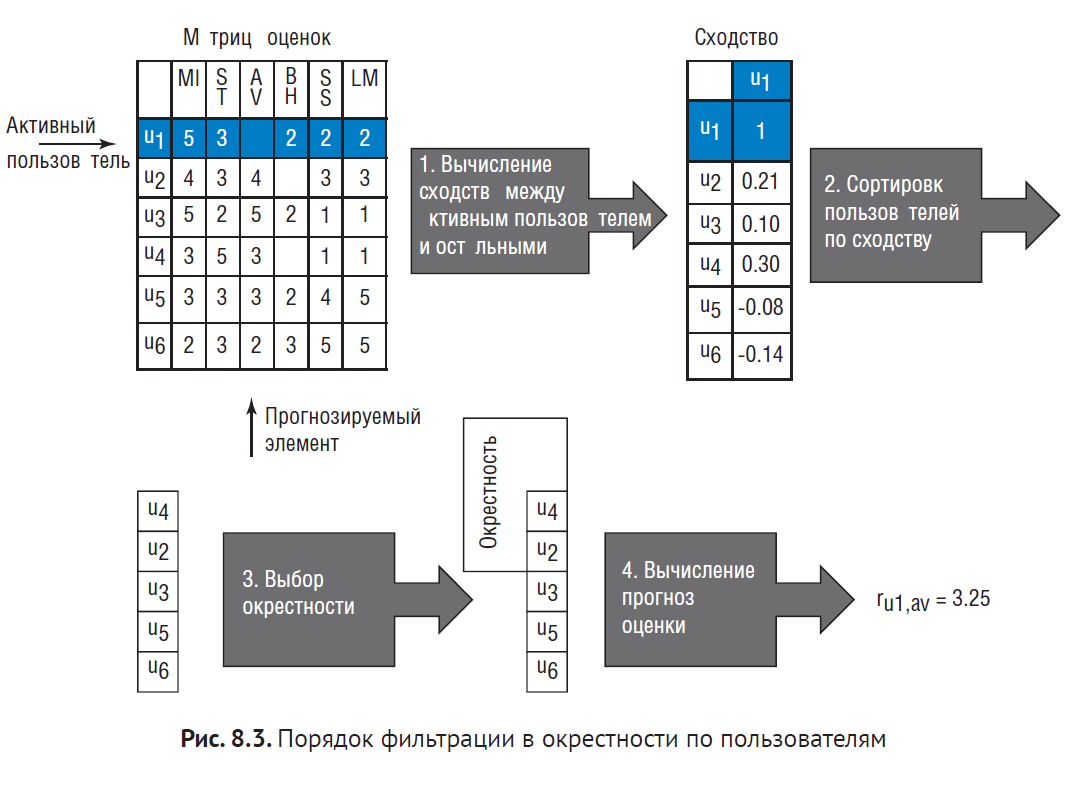
MSE дает большие штрафы за большие ошибки и таким образом одна большая ошибка будет значить гораздо больше, чем несколько маленьких. В то же время при использовании MAE большие ошибки в меньшей степени влияют на результат. Если нужно, чтобы ни один из пользователей не получал плохие рекомендации, нужно использовать MSE. Но если вы понимаете, что всем не угодишь, то, вероятно, достаточно использовать MAE.

Показатели Top-N с учетом ранжирования (MRR, nDCG)

Топ-N число соседей данного пользователя или элемента, мы можем найти похожий товар или похожих пользователей исходя из его соседей.

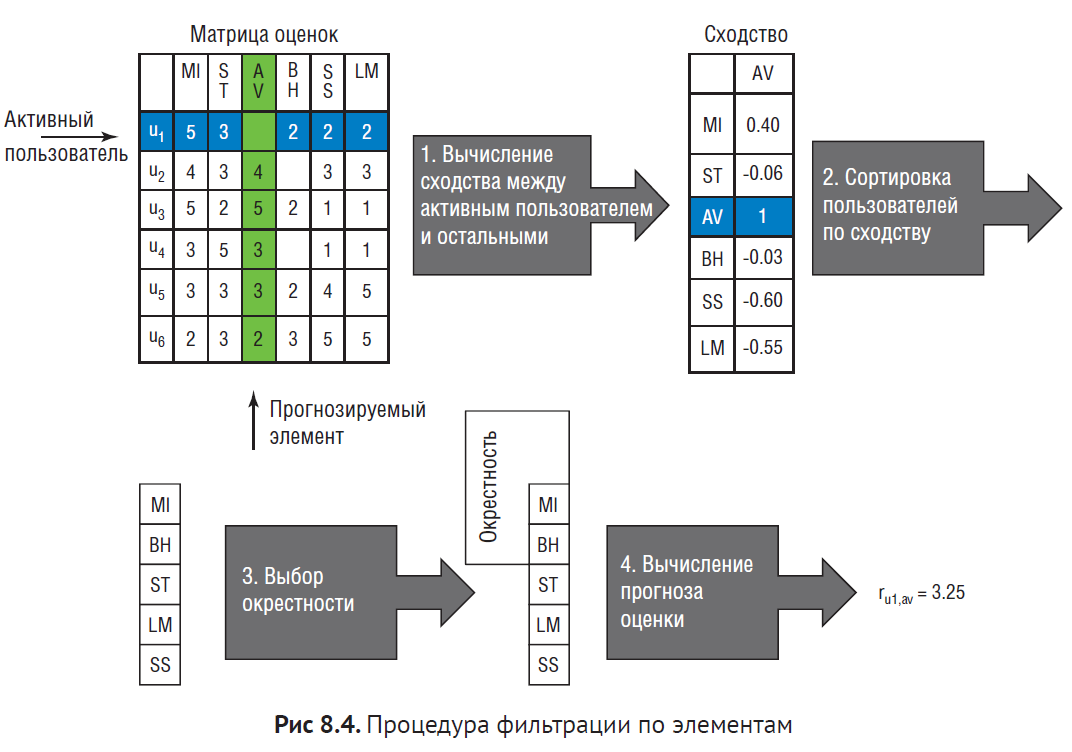
Допустим мы должны дать пользователю 5 рекомендаций. Пользователь высоко оценил 3 фильма. Для каждого из оценённых фильмов мы можем найти 3 соседених фильмов. Итого у нас есть 9 фильмов для рекомендации (возможно не уникальных). Как выбрать 5 из 9?

Пользовательская коллаборативная фильтрация.



1. Вычисление сходства между активным пользователем и остальными
2. Сортировка пользователей по сходству
3. Выбор окрестномти
4. Вычисление прогноза оценки

Коллаборативная фильтрация по элементам.



1. Вычисление сходства между выбранным элементом и остальными
2. Сортировка пользователей по сходству
3. Выбор окрестности
4. Вычисление прогноза оценки

Экспертные рекомендательные системы

Экспертные рекомендательные системы формируют блок рекомендаций

на основе ассоциативных правил, составленных вручную специалистами.

Эти системы рекомендуют хорошие вина, книги и подобные вещи и при-

меняются в областях, где, как правило, необходимо быть экспертом, чтобы

давать советы.

Однако дни экспертных систем практически на исходе, поэтому сейчас па-

раметр чьи мнения утратил свою актуальность. Почти все сайты опираются

на мнение большинства. Исключения: рекомендация вина.

Проблема холодного старта в рекомендательных системах.

Для совместной фильтрации требуются данные, которые трудно получить при

появлении новых пользователей и новых элементов. В итоге у вас нет данных

для выработки рекомендаций. Опять же, чтобы обойти эту проблему, можно

попросить новых пользователей оценить несколько элементов сразу при их по-

явлении. Или, как вариант, можно дать новым пользователям посмотреть но-

вый контент, многим это понравится.

Также необходимо рекламировать новые элементы чтобы они попадали в рекомендации.

Какие методы можно использовать для заполнения пропущенных данных

Часто вы будете сталкиваться с ситуациями, когда рекомендатор не выдает

рекомендаций вообще или дает их мало. Это может быть проблемой во время оценки системы, поэтому можно заполнить пробелы данных с помощью простого алгоритма, выбирая наиболее популярные элементы, среднюю оценку

для данного элемента или среднюю оценку пользователей. Несколько более

сложным решением является использование базовых рекомендателей, кото-

рые мы рассмотрим в главе 11. Эта тема также кратко обсуждается в статье,

которую мы рекомендовали в предыдущем разделе.

Базисным предиктором, который упрощает добавление значений в пустые ячейки матрицы. Хотя их можно использовать в качестве рекомендательный системы, здесь это будет способ сделать матричную факторизацию лучше.

Изображение выглядит как текст

Автоматически созданное описание

Что такое векторизация TF/IDF

Изображение выглядит как текст

Автоматически созданное описание

Изображение выглядит как текст, человек, снимок экрана, документ

Автоматически созданное описание

Изображение выглядит как текст

Автоматически созданное описание

Типы рекомендательных систем

Совместная рекомендательная система (пользователь-пользователь)

Контекстная рекомендательная система

Система на основе знаний

Гибридная рекомендательная система:

Монолитная

Смешанная

Ансамбль

Неперсонализированные рекомендации.

**Неперсонализированные** **рекомендации** — **рекомендации** без учета личных предпочтений пользователя, когда система сортирует по средней оценке рейтинга популярные товары или фильмы.

Неперсонализировнные рекомендации – это топ 10, а также выборочные рекомендации, вместе с этим покупают то.

Коэффициент корреляции Пирсона на коэффициент Отиаи.

Изображение выглядит как текст

Автоматически созданное описание

Формула для Пирсона и Отиаи выглядит похоже.

При расчёте среднего 0 не учитываются.

При вычитании среднего в Пирсоне необходимо чтобы элемент не был нулём у обоих пользователей, У Отиаи только у одного.

Совместная фильтрация и матрица оценок.

Простыми словами, совместная фильтрация выводит вам список рекомендуемых элементов. Список создается на основе данных о людях, чьи вкусы схожи с вашими и которые видели что-то, чего еще не видели вы.

Совместная фильтрация (CF, Collaborative filtering):

1. Item-Item
2. User-user

То есть мы находим пользователей, схожих с активным пользователем, а за-

тем рекомендуем ему фильмы, которые понравились схожим пользователям.

И второй способ – мы находим элементы, схожие с теми, которые пользователю понравились ранее. Чтобы сделать все это, нам снова понадобятся матрицы оценок, где описаны предпочтения пользователей.

Процедура совместной фильтрации (пользователь-пользователь):

1. Вычисляем сходство между текущим пользователем и остальными (коэффициент Пирсона, Отиаи или косинусное расстояние)
2. Сортируем пользователей по сходству
3. Выбираем окрестность (k-mean,топ-N, пороговое значение)
4. Делаем прогноз

Процедура совместной фильтрации (элемент-элемент):

1. Вычисляем сходство между текущим элементом и остальными (коэффициент Пирсона, Отиаи или косинусное расстояние)
2. Сортируем элементы по сходству
3. Выбираем окрестность
4. Делаем прогноз

Примеры успешной реализации бизнес-моделей, основанных на использовании рекомендательных систем.

Youtube

Недостатки неперсонализированных и слабо-персонализированных систем.

Плюсы:

**•** легко добавить новые элементы. Создайте вектор признаков эле-

мента и – вперед!

**•** большого количества трафика не нужно. Поскольку вы можете вы-

числить сходство на основе описаний контента, вы можете форми-

ровать рекомендации с первого посещения или первой оценки;

**•** рекомендации не подвержены отклонениям, связанным с популяр-

ностью. Непопулярный фильм и популярный попадают в выдачу

с одинаковым шансом.

 Минусы:

**•** оценки приобретают большую силу. Если вам нравится фантастика

с Харрисоном Фордом, система выдаст вам фильмы с Харрисоном

Фордом, даже если это не фантастика;

**•** выдача строго определена, без сюрпризов;

**•** ограниченное понимание содержания. Затруднительно учесть все

признаки контента, которые нравятся пользователю, т. е. система

может легко неправильно понять его вкусы.

Изображение выглядит как текст

Автоматически созданное описание

Неперсонализированные саммые популярные

Слабо-персонализированые: контекстная фильтрация

Персонализированные: своместная фильтрация, гибридная

Способы получения и обработки информации о предпочтениях пользователя.

1.Явные оценки – попросить пользователя дать несколько оценок.

2. Неявные оценки – программное отслеживание пользователя: перемещение курсора мыши, направление взгляда, история поиска, анализ соц.сетей.

Обработка: Каждому действию добавляется коэффициент и вычисляется предполагаемая оценка пользователя элементу, можно использовать временное затухание чтобы начислять новым элементам больше баллов, также можно добавлять баллы редкими элементам, потому что они лучше показывают вкусы пользователя.

Алгоритм Slope One.

Изображение выглядит как текст, человек, снимок экрана

Автоматически созданное описание

Изображение выглядит как текст

Автоматически созданное описание

Изображение выглядит как текст

Автоматически созданное описание

Изображение выглядит как текст, внутренний, снимок экрана

Автоматически созданное описание

Реализация рекомендательной системы на основе подбора элементов

*Контентная фильтрация* опирается на метаданные объектов из вашего ка-

талога. Например, Netflix пользуется описаниями фильмов. В зависимости от

алгоритма система может формировать рекомендации путем подбора объек-

тов, аналогичных тем, которые понравились пользователю ранее, путем со-

поставления объектов с данными из пользовательского профиля, либо, если

пользователь не задействован, путем поиска схожих объектов из всего кон-

тента. При наличии пользовательского профиля система анализирует каж-

дый профиль, в котором указаны категории контента. Если бы сервис Netflix

применял контекстную фильтрацию, он мог бы составлять пользовательские

профили с разбивкой по жанрам – например, триллеры, комедии, драмы и

новинки – и указывать рейтинг каждого из жанров. В этом варианте фильм

попадает в рекомендации, только если его рейтинг соответствует рейтингу,

указанному пользователем.

Метод k-средних. Как выбрать число k для алгоритма кластеризации

Метод локтя

Метод силуэтов

Подбором

Изображение выглядит как текст

Автоматически созданное описание

Изображение выглядит как текст

Автоматически созданное описание

Изображение выглядит как текст

Автоматически созданное описание

Изображение выглядит как текст

Автоматически созданное описание

Как работает алгоритм Funk SVD

У каждого пользователя есть оценки-выбросы, они мешают понять его истинные вкусы.

Истинные вкусы – это как линия на графика, а оценки пользователя это облако точек. Отбросив часть точек мы получим линию.

Основная идея за-

ключается в том, что, глядя на данные о поведении пользователей, вы можете

найти категории или темы, которые могут объяснить вкусы пользователей менее

точно, чем по единичным фильмам, но более точно, чем по крупным жанрам.

Эти факторы позволяют нам расположить подобные фильмы ближе друг к другу.

Алгоритм позволяющий сделать это называется SVD.

SVD возвращает матрицу U (вкусы пользователей,матрицу сигма необходимую для отброски шумов, матрицу vt – обозначающаю профили элементов.

На вход SVD принимает матрицу оценок M.

print(U\_reduced[Sara]\*Vt\_reduced[:,AceVentura])

Новый пользователь

u\_kim = r\_kim \*Vt\_reduced.T\* inv(Sigma\_reduced)

print(u\_kim\*Vt\_reduced[:,AceVentura])

## FunkSVD

У алгортима SVD есть два серьёзных минуса:

1. Он медленный, а значит чтобы применить к реальной матрице пользователь-элемент в которой будут тычсичи (если не миллионы) строк и колонок, будет почти невозможно.
2. Проблема разряженности, в реальности у каждого пользователя будет максимум 1% оцененных товаров, и заполнять остальные товары среднеми значениями (используя даже продвинутую технику базисных предикатов) не очень правильно.

Чтобы решить эту проблем был разработан FunkSVD. FunkSVD - принимает на вход не матрицу пользователей-товаров.А список всех оценок, то есть он принимает только выставленные оценки, и ему не нужно заполнять пропуски. Во вторых мы работаем со списком, а не с бесконечной матрицей, это должно повысить производительность.

Метод FunkSVD возвращает: общее смещение, смещение пользователя, смещение элемента, факторы пользователей (q) и факторы элементов (p)

Прогноз оценки рассчитывается так:

Изображение выглядит как текст, антенна

Автоматически созданное описание

Pi – это факторы элементы i

Qu – это факторы пользователя u

Pi\*qu – это число

Bu – смещение пользователя

Bi – смещение элемента

μ – общее смещение

Метод FunkSVD,

Выполняет градиентный спуск чтобы найти факторы пользователя и элементов. Каждый новый фактор рассчитывается на основе всех предыдущих.

Метод SVD++ это FunkSVD использующий также неявный отклик пользователя.

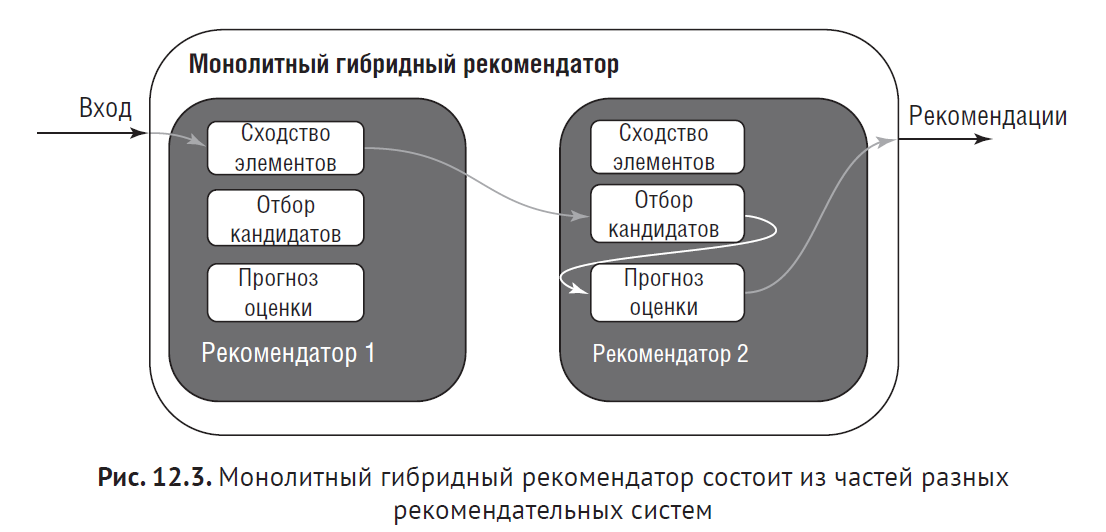
Как работает алгоритм SVD++

Постановка задачи рекомендательной системы

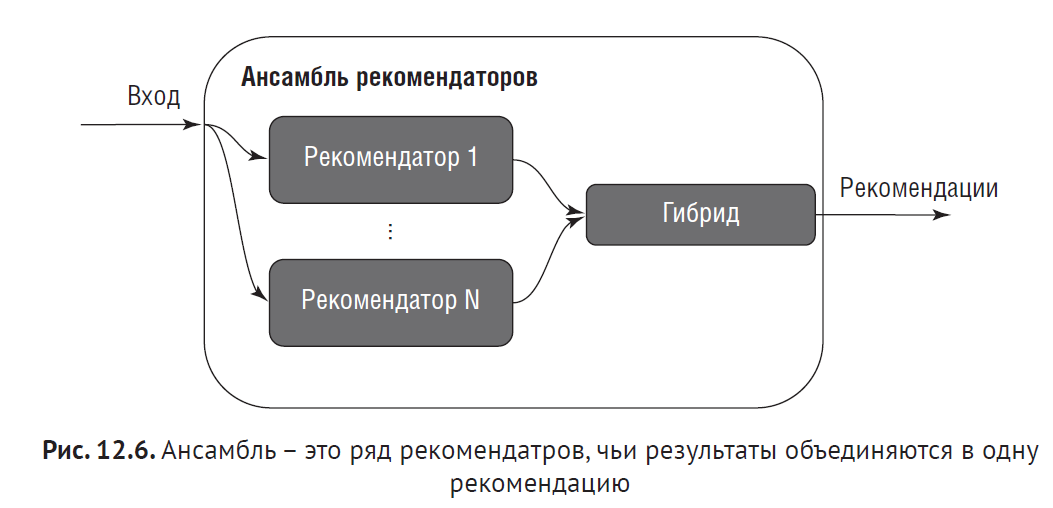
Гибридные рекомендательные системы

Изображение выглядит как текст

Автоматически созданное описание







SVD-разложение матрицы градиентным спуском

Изображение выглядит как текст

Автоматически созданное описание

Решение проблемы холодного старта алгоритмами

Мы получаем демографические данные о пользователе через мета данные запроса. С помощью API социальных сетей мы можем узнать уровень образования, социальный статус и прочие характеристики.

Существуют два основных подхода к применению в рекомендациях демографической информации о пользователе:  
— Экспертным образом составляются стереотипы для различных демографических категорий.

— Демографические категории определяются автоматически путем выявления кластеров пользователей со схожими интересами. Рекомендации строятся на основе того, какие рейтинги проставляли пользователи из той же категории, то есть того же возраста, пола, местоположения и т. д.

Для кластеризации по демографическим данным естественно использовать метод [*k*-средних](http://ru.wikipedia.org/wiki/K-means), так как в этом случае каждый кластер определяется точкой своего центра и, в следствии этого, хорошо интерпретируется.

После того, как кластеры пользователей получены и для каждого нового пользователя, указавшего свои демографические данные, мы знаем кластер, к которому он относится, мы можем улучшить рекомендации на холодном старте с помощью групповых рекомендаций или фильтр-ботов.

Наиболее естественным является метод **групповых рекомендаций (group recommendation to individual user)**, название которго говорит само за себя: мы подбираем новому пользователю такие рекомендации, которые нравятся большинству пользователей в его демографической категории.

Альтернативным подходом являются **фильтр-боты (filterbots)**, которые генерируют дефолтные рейтинги для нового пользователя. То есть при регистрации фильтр-боты автоматически сгенерируют несколько рейтингов для пользователя на основе его демографических данных, которые будут использованы алгоритмами коллаборативной фильтрации на холодном старте.

Холодный старт для элементов

Алгоритм **relevance feedback** составляет профиль тегов (то есть ключевых слов) для каждого пользователя по контент-профилям страниц, которые лайкал этот пользователь.  
Новые элементы рекомендуются тем пользователям, прифили тегов которых наиболее коррелируют с контент-профилем вновь добавленного элемента.

Альтернативой может быть алгоритм LDA. Это алгоритм кластеризации – который разбивает все элементы на кластеры, после этого выясняется предпочтения пользователей по этим кластерам. Новый элемент добавляется в существующий кластер и рекомендуется тем пользователям, которые предпочитают данный кластер.

Алгоритм кластеризации k-средних

Алгоритм кластеризации k-средних находит такие точки k, называемые цен-

троидами, которые удовлетворяют утверждению о том, что сумма расстояний

между всеми элементами и присвоенными ими центроидами будет настолько

мала, насколько это возможно. Пока пишу этот текст, смотрю как на участке

работает поливалка для цветов. Задача примерно похожая – надо крепко подумать о том, где разместить поливалку, потому что вода в Дании дорогая, и нужно найти такую точку, чтобы поливать как можно больше цветов, а воды тратить

поменьше. Кластеризация k-средних подойдет и для этой задачи, так как этот

алгоритм сгенерирует центроиды, в которые можно разместить поливалку.

Алгоритм выполняет следующие этапы:

1. Выбирается k центров кластеров.
2. Дальше в цикле делается следующее:
   1. для каждой точки данных в наборе находится центроид с кратчай-

шим расстоянием;

* 1. когда все точки будут назначены центроиду, вычисляет сумму всех

расстояний между элементом и его центроидом.

1. Если расстояние оказалось не меньше, чем в предыдущей итерации,

возвращаются кластеры.

1. Каждый центроид перемещается в центр назначенного кластера.

Существует много способов выбрать начальные положения центроидов,

и это очень важный шаг, от которого сильно зависят результаты всей кла-

стеризации.