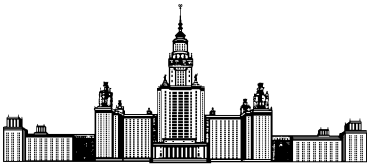
Московский государственный университет имени М. В. Ломоносова
Факультет вычислительной математики и кибернетики
Кафедра математических методов прогнозирования



ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА

«Кластеризация сессий в общих учетных записях»

Выполнил:
студент 4 курса 417 группы
Борисов Алексей Антонович

Научный руководитель:
д. ф.-м. н., профессор
Дьяконов Александр Геннадьевич

Москва, 2023

Содержание

1 Введение **4**

1.1 Обзор литературы 6

1.2 Постановка задачи 8

2 Предлагаемый подход **9**

2.1 Общая модель 9

2.2 Описание предметной области 10

2.3 Базовое решение 11

2.4 One-Step-ALS 13

2.5 Взвешивание действий 14

2.6 Кластеризация с использованием априорной информации 15

3 Эксперименты **17**

3.1 Синтетические данные 18

3.2 Разметка детский/взрослый 18

4 Optional: Детекция совместных аккаунтов **18**

5 Заключение **18**

6 Список литературы **18**

Аннотация

Рекомендательные системы являются неотъемлемой частью современного мира, они позволяют среди огромного количества сущностей выбрать наиболее релевантные для конкретного человека. Однако нередко рекомендательной системой с одного аккаунта и/или устройства пользуются одновременно несколько человек. Это может сильно ухудшать качество рекомендаций, так как большинство подходов предполагают, что каждому аккаунту соответствует один реальный пользователь. Целью данного исследования является описание подхода для кластеризации сессий пользовательской активности в совместных аккаунтах, а также проведение необходимых экспериментов для оценки качества представленного подхода. Одной из проблем исследований по данной теме является отсутствие необходимой разметки, однако особенностью данной работы является использование разметки принадлежности действий реальным пользователям на основе распознавания голоса в голосовом помощнике «Маруся». Также поведение пользователя в рекомендательной системе описывается не на уровне отдельных объектов, а на уровне действий, которые пользователь совершает в рекомендательной системе. Такой подход должен лучше отражать желания пользователя. Общая модель, которая представлена в данной работе может быть использована для осуществления кластеризации сессий в произвольной рекомендательной системе.

1 Введение

В современном мире нас окружает огромное количество стриминговых сервисов, например, «Youtube» и «Кинопоиск» — онлайн-сервисы видео, «Яндекс Музыка» и «VK Музыка» — онлайн-сервисы музыки. Во многих таких сервисах распространено явление, что одним аккаунтом в действительности пользуются несколько человек. Это связано прежде всего с тем, что внутри одной семьи все пользуются сервисом из-под одного аккаунта. Особенно характерно такое использование голосовых помощников: в доме находится одна умная колонка с голосовым помощником, через которую может осуществляться использование стриминговых сервисов, например, онлайн-кинотеатра или сервиса для прослушивания музыки.

У подавляющего большинства стриминговых сервисов имеется своя рекомендательная система, которая предоставляет для каждого пользователя персонализированные релевантные предложения, будь то фильмы или музыка. Однако рекомендательные системы зачастую предполагают, что каждому аккаунту соответствует один реальный пользователь, и возможна ситуация, что рекомендации для совместных аккаунтов будут релевантны только для одного из пользователей, или вовсе не будут релевантны никому.

Для борьбы с проблемами, связанными с совместными аккаунтами, необходимо уметь разделять активность внутри аккаунта по реальным пользователям. В работе Jiang et al. [5] выделяют три основные задачи:

1. По истории сессий аккаунта определить набор реальных пользователей.
2. По заданной новой сессии определить, какому пользователю из заранее определенного множества она принадлежит.
3. Использование информации о принадлежности сессий разным пользователям для улучшения качества рекомендаций.

Данное исследование сосредоточено на задаче 1, кластеризации сессий пользовательской активности. Ниже в текущем разделе описаны существующие подходы к обработке совместных аккаунтов в рекомендательных системах, а также описывается формальная постановка задачи. В разделе 2 описана общая модель для решения поставленной задачи, а также рассмотрены различные варианты реализации пред-

ставленной модели. Далее в разделе 3 проанализировано качество рассмотренных решений на реальных данных стримингового сервиса музыки в голосовом помощнике «Маруся». Работы по данной теме сталкиваются с принципиальной проблемой отсутствия необходимой разметки о принадлежности действий конкретным пользователям, что мешает реально оценить качество описанных решений, однако в настоящей работе использована разметка, основанная на распознавании голоса.

1.1 Обзор литературы

Исследований по работе с совместными аккаунтами в рекомендательных системах не так много, так как тема является довольно специфичной.

В наиболее ранней из рассмотренных работ, Zhang et al. [18] предлагают представлять совместные аккаунты в виде объединения линейных подпространств, где каждому отдельному пользователю соответствует одна гиперплоскость. Выписывается невыпуклая оптимизационная задача, которую удается решить при помощи EM-алгоритма или обобщенного метода главных компонент (GPCA). Кроме собственно модели, авторы разработали статистический тест, который позволяет оценить вероятность того, что полученный результат является случайным. В этой работе использовался датасет CAMRa2011 [1], обладающий разметкой для 290 совместных аккаунтов, а также были сформированы синтетические совместные аккаунты на основе Netflix Dataset.

Существует ряд статей, которые улучшают качество рекомендаций на случай совместных аккаунтов без явного поиска разбиения аккаунта по пользователям. Например, в работе Bajaj et al. [2] для улучшения качества рекомендаций в сервисе онлайн телевидения предлагается следующий подход: на основе всей имеющейся статистики просмотров формируются кластеры каналов, после чего с помощью алгоритма Apriori определяются частые комбинации кластеров, которые представляют собой некототрые профили пользователей (Personas). Далее для каждого аккаунта выделяется набор соответсвующих ему профилей (возможно пересекающихся), которые впоследствии используются при формировании рекомендаций. Схожий подход использовали в работе Verstrepen и Goethals [12] для улучшения качества коллаборативной фильтрации на случай совместных аккаунтов. Вместо того чтобы определять оценку нового объекта i как сумму его похожестей на объекты S уже оцененные пользователем:

$$s_{IB}(S, i) = \sum_{j \in S} sim(j, i) \cdot |KNN(j) \cap \{i\}|$$

предлагается оценивать новый объект по подможеству объектов дающему наибольшую оценку и при этом штрафовать за размер использованного подмножества:

$$s_{DAMIB}(S, i) = \max_{D \in 2^S} \frac{1}{|D|^p} \cdot s_{IB}(D, i), \quad p \in [0, 1]$$

Этот подход позволяет избежать ситуации, в которой рекомендации не релевантны никому из пользователей аккаунта из-за того что они имеют разные вкусы.

Отдельно можно выделить исследования, использующие графовые нейронные сети для выявления и обработки совместных аккаунтов. В работе Wen et al. [16] для каждого аккаунта строится ориентированный граф сессий, вершины в котором обозначают объекты v_i, с которыми взаимодействовал пользователь, а наличие ребра $v_i \to v_j$ означает что в одной из сессий объект v_j шел после v_i. Векторное представление сессии строится с помощью усреднения векторных представлений объектов сессии. Далее для определения векторных представлений реальных пользователей используется механизм внимания (attention). В другой работе Jiang et al. [5] строят уже один неориентированный граф, содержащий вершины объектов, а также мета-информации, например, вершины исполнителей или альбомов. Векторные представления объектов формируются при помощи алгоритма на основе случайного блуждания. Для получения векторного представления сессии используется эвристическое взвешенное среднее, для того чтобы уменьшить влияние объектов, с которыми во время сессии взаимодействовали несколько раз. Далее для кластеризации сессий используется алгоритм Affinity Propagation.

В большинстве исследований, связанных с обработкой совместных аккаунтов, качество решения оценивается на задаче рекомендаций при помощи классических метрик ранжирования: Recall@n, Precision@n, MRR, NDCG и др. [2, 12, 7, 16, 14, 19] В тех работах, где пытаются оценить качество кластеризации действий из-под пользовательского аккаунта, зачастую используют синтетические данные, так как в открытом доступе практически отсутствуют необходимые датасеты с наличием разметки в совместных аккаунтах [18, 5]. Единственный найденный датасет с необходимой разметкой, CAMRa2011 [1], обладает такой разметкой только для 290 аккаунтов, чего недостаточно для оценки качества решений. Поэтому для оценки качества кластеризации используются синтетические данные, в которых аккаунты представляют собой объединение действий из 2 и более реальных аккаунтов.

Особенностью данного исследования является использование реальной разметки принадлежности действий в совместных аккаунтах, полученной на основе распознавания голоса в голосовом помощнике «Маруся». Также в настоящей работе взаимо-

действие пользователя с рекомендательной системой рассматривается не на уровне объектов, а на уровне действий, что должно лучше отражать желания реального пользователя. Общий подход, который будет представлен далее может быть использован для осуществления кластеризации сессий в произвольной рекомендательной системе.

1.2 Постановка задачи

Для начала введем ряд обозначений:

U — множество пользовательских аккаунтов.

P_u — множество реальных людей, пользующихся рекомендательной системой из-под аккаунта $u \in U$.

A — множество действий, которое может совершать пользователь внутри рекомендательной системы.

$s = [a_1, \ldots, a_m]$ — сессия, последовательность действий внутри рекомендательной системы, выполненная одним человеком, $a_i \in A$. Зачастую разумно считать, что если действия были выполнены через незначительный промежуток времени, то они были совершены одним человеком.

S_u — множество сессий, совершенных из-под аккаунта $u \in U$.

Задача заключается в кластеризации сессий, совершенных из-под одного аккаунта. Решением задачи является алгоритм f, который для множества сессий $S_u = \{s_1^u, \ldots, s_N^u\}$ выдает принадлежность каждой сессии одному из K кластеров: $\{(s_1^u, k_1), \ldots, (s_N^u, k_N)\}$, где $k_i \in \{0, \ldots, K\}$. Каждому кластеру должен соответствовать реальный человек $p \in P_u$.

Таким образом, естественными функционалами качества в данной задаче являются внешние меры оценки качества кластеризации, использующие информацию о распределении объектов по кластерам.

2 Предлагаемый подход

В данном разделе вначале будет описана общая модель, которая представляет собой последовательность действий, которые решают поставленную задачу. Далее будет описана предметная область музыкального сервиса и представлен простейший вариант модели, после чего описаны улучшения, которые позволяют повысить качество.

2.1 Общая модель

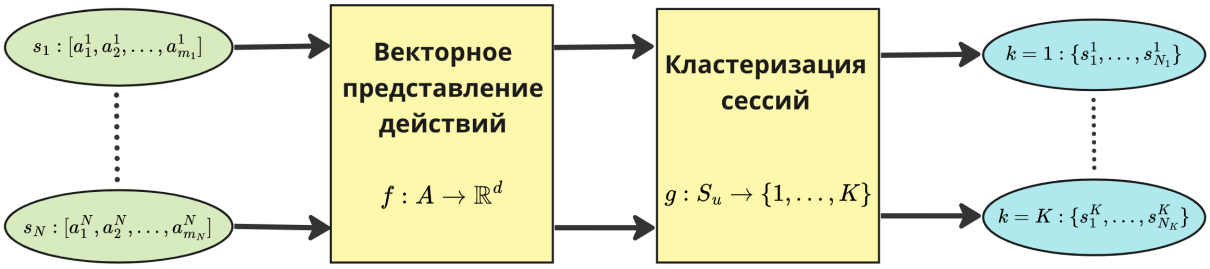


Рис. 1: Общая модель кластеризации сессий одного аккаунта.

На рисунке 1 изображена общая модель кластеризации сессий одного аккаунта, подробнее остановимся на каждом элементе данной схемы.

На вход поступает множество сессий, каждая из которых представляет собой последовательность некоторых действий в рекомендательной системе. Далее для каждого действия строится векторное представление. Несмотря на то, что действия могут быть разного вида, все они должны отображаться в одно векторное пространство, так как следующий шаг работы модели заключается в кластеризации сессий на основе близости векторных представлений действий. Здесь можно выделить три разных подхода:

1. Построение векторного представления сессии на основе векторных представлений действий и дальнейшая кластеризация в пространстве сессий.
2. Кластеризация в векторном пространстве действий с последующим определением кластера для каждой сессии, например, при помощи голосования среди действий из сессии.

3. Кластеризация в векторном пространстве действий с использованием дополнительной априорной информации о принадлежности групп объектов к одному кластеру (действия, относящиеся к одной сессии, обязаны лежать в одном кластере).

2.2 Описание предметной области

Для последующего рассмотрения более конкретных вариантов применения общей модели удобно остановиться на определенной предметной области и описывать предложенные подходы в приложении к ней. Они могут быть естественным образом адаптированы к использованию в других предметных областях. В этой работе рассматривается предметная область для стримингового сервиса музыки, интегрированного в голосовой помощник. Примерами таких голосовых помощников являются Алиса от компании Яндекс и Маруся от компании VK.

Основные сущности: трек, исполнитель и плейлист. Трек — аудиозапись, которая может быть воспроизведена для пользователя. У каждого трека есть информация о его исполнителе. Плейлист — заранее подготовленный список треков, например, в большинстве сервисов понравившиеся треки можно добавить в плейлист «Моя музыка».

В рассматриваемой модели музыкального сервиса выделим три типа действий, которые может совершать пользователь:

- Воспроизведение определенного трека.
  Пример: «Включи We Will Rock You».
- Воспроизведение треков определенного исполнителя.
  Пример: «Включи Pink Floyd».
- Воспроизведение треков определенного плейлиста.
  Пример: «Включи мою музыку», «Включи классику русского рока».

Необходимо предоставить отображение $f : A \to \mathbb{R}^d$, которое каждому действию ставит в соответствие векторное представление. Как уже было упомянуто, все представления должны лежать в одном векторном пространстве.

2.3 Базовое решение

Основным понятием является трек, поэтому сперва рассмотрим варианты получения векторных представлений для треков. Далее они могут быть использованы как основа для построения векторных представлений исполнителей и плейлистов.

Векторные представления треков можно получить, например, при помощи низкоранговых матричных разложений, опишу два основных типа:

- user-item разложение «пользователь на трек», где элементы матрицы отражают факт того, что из-под конкретного пользовательского аккаунта был прослушан конкретный трек, возможно с учетом количества прослушиваний.
- item-item разложение «трек на трек», где элементы матрицы отражают количество сопрослушиваний треков в стриминговом сервисе.

В общем виде задачу низкорангового разложения матрицы можно записать в следующем виде:

$$A_{n\times m} \approx W_{n\times k} H^T_{m\times k}$$

Если матрица A представляет собой матрицу взаимодействий n пользовательских аккаунтов с m треками, то строки матрицы W представляют собой векторные представления для пользовательских аккаунтов, а строки матрицы H представляют собой векторные представления для треков.

Если матрица A представляет собой матрицу сопрослушиваний треков между собой (матрица A все равно может являться прямоугольной, так как множества треков слева и справа в разложении могут не совпадать), то строки матриц W и H представляют собой вектора треков, однако в разных векторных пространствах.

Существует большое количество различных способов задать функционал, оптимизируя который можно получить качественное низкоранговое матричное разложение, например:

- Классический квадратичный функционал:

$$\mathscr{L}(W,H) = \sum_{(u,i)\in R} (\langle w_u, h_i\rangle - r_{ui})^2 + \lambda\|W\|^2 + \mu\|H\|^2$$

- Weighted Regularized Matrix Factorization (WR-MF) [4]:

$$\mathscr{L}(W,H) = \sum_{u,i} c_{ui}(\langle w_u, h_i\rangle - 1)^2 + \lambda\|W\|^2 + \mu\|H\|^2$$

- Maximum Margin Matrix Factorization (MMMF) [15]:

$$\mathscr{L}(W,H) = \sum_{(u,i,j)\in D} \max(0, 1 - \langle w_u, h_i - h_j\rangle) + \lambda\|W\|^2 + \mu\|H\|^2$$

- Bayesian Personalizing Ranking (BPR) [10]:

$$\mathscr{L}(W,H) = \sum_{(u,i,j)\in D} \ln\sigma(\langle w_u, h_i - h_j\rangle) + \lambda\|W\|^2 + \mu\|H\|^2$$

Представленные функционалы можно оптимизировать, например, при помощи стохастического градиентного спуска (SGD), а также в некоторых частных случаях возможно использование метода чередующихся наименьших квадратов (ALS).

Использование разложения любого из этих типов позволяет получить векторные представления треков, которые можно использовать как векторное представление действия «воспроизведение трека».

Рассмотрим простейший способ получения векторного представления для исполнителя, которое можно использовать как векторное представление для действия «воспроизведение треков исполнителя». Этот способ заключается в усреднении векторных представлений треков исполнителя:

$$e_{\text{artist}} = \frac{1}{|T_{\text{artist}}|} \sum_{t\in T_{\text{artist}}} e_t, \quad \text{где } T_{\text{artist}} \text{ — множество треков исполнителя artist.}$$

Аналогичным образом можно получить векторное представление для плейлиста, которое можно использовать как векторное представление для действия «воспроизведение треков плейлиста»:

$$e_{\text{playlist}} = \frac{1}{|T_{\text{playlist}}|} \sum_{t\in T_{\text{playlist}}} e_t, \quad \text{где } T_{\text{playlist}} \text{ — множество треков в плейлисте playlist.}$$

В базовом решении каждое действие рассматривается в качестве отдельной сессии, то есть фактически время между действиями никак не учитывается. Таким образом, в качестве векторного представления сессии можно использовать векторное представление соотвествующего единственного действия из которого состоит сессия.

Далее получить разбиение сессий по пользователям можно применив к множеству векторных представлений пользователей произвольный алгоритм кластеризации, например, K-Means.

2.4 One-Step-ALS

Рассмотрим подход, позволяющий получить более качественное векторное представление для набора треков S, чем простое усреднение. Этот подход можно будет использовать для получения векторных представлений исполнителя, плейлиста, а также для формирования векторного представления сессии по векторным представлениям действий.

Фактически, представленный способ с усреднением векторов, заключается в минимизации следующего функционала:

$$\mathscr{L}(S) = \sum_{t \in S} (e_s - e_t)^2 \longrightarrow \min_{e_s}$$

Предлагается воспользоваться другим функционалом, который уже упоминался при описании способов матричного разложения:

$$\mathscr{L}(S) = \sum_{t \in S} (\langle e_s, e_t \rangle - r_t)^2 + \lambda \|e_s\|^2 \longrightarrow \min_{e_s}$$

Для представленной задачи минимизации можно записать аналитическое решение:

$$\frac{\partial}{\partial e_s} \mathscr{L}(S) = 2 \sum_{t \in S} (\langle e_s, e_t \rangle - r_t) \cdot e_t + 2\lambda \cdot e_s = 2 \cdot (\sum_{t \in S} e_t e_t^T + \lambda \cdot I) \cdot e_s - 2 \sum_{t \in S} r_t e_t = 0$$

$$\hat{e}_s = \min_{e_s} \mathscr{L}(S) = (\sum_{t \in S} e_t e_t^T + \lambda \cdot I)^{-1} \cdot \sum_{t \in S} r_t e_t$$

Можно использовать различные значения r_t, чтобы дать некоторым векторам большее влияние на итоговый вектор набора. Например при формировании векторного представления исполнителя можно давать популярным трекам больший вес, так как они могут лучше характеризовать исполнителя. Если необходимо учитывать все объекты с одинаковым весом, то можно положить все r_t равными единице. Далее в разделе 2.5 будет представлен другой альтернативный способ учесть важность отдельных элементов набора S.

Естественным названием для данного подхода является One-Step-ALS, так как фактически получение векторов для наборов треков заключается в применении

одного шага алгоритма ALS к матрице A «наборы треков на треки», где в строке соответствующей набору треков S стоят значения r_t в столбцах соответствующих трекам $t \in S$.

Оценим вычислительную сложность обоих представленных подходов для формирования векторного представления для набора векторов. Используем следующие обозначения: N — количество векторов в наборе, d — размерность векторов. Подход с усреднением векторов имеет сложность $O(N \cdot d)$. При использовании One-Step-ALS необходимо вычилсять N матриц вида $e_t e_t^T$, а также вычислять обратную матрицу, поэтому итоговая вычислительная сложность данного подхода — $O(N \cdot d^2 + d^3)$.

В представленной работе d это размерность векторов из низкорангового матричного разложения, поэтому d не превосходит нескольких сотен. Количество треков исполнителя, треков в плейлисте или действий в сессии также в подавляющем числе случаев не превосходит нескольких сотен. Таким образом, в данном случае сложность $O(N \cdot d^2 + d^3)$ не представляет больших проблем, тем более что, например, вектора для исполнителей и плейлистов могут быть посчитаны заранее. Однако при использовании One-Step-ALS для формирования вектора сессии по векторным представлениям действий это может стать узким местом модели.

2.5 Взвешивание действий

Одним из предложенных способов определения кластера для сессии является кластеризация действий пользователя с последующим голосованием внутри каждой отдельной сессии. При таком подходе, все действия имеют одинаковое влияние на определение итогового кластера, однако в силу того что действия могут иметь разную природу, некоторые из них могут являться намного более важными чем другие. Пример: в онлайн магазине пользователь может совершать следующие действия: «открыть страницу с товаром», «положить товар в корзину», «купить товар». Очевидно что покупка является более важным действием чем просмотр товара. Решением данной проблемы является использование взвешенного голосования.

Пусть имеются действия n типов: $A_1, \ldots, A_n$ с весами $w_1, \ldots, w_n$,
где $A_i \subseteq A, w_i \geq 0, \ \forall i = \overline{1,n}, \quad A_i \cap A_j = \emptyset, \ \forall i \neq j$. Тогда для сессии $s = [a_1, \ldots, a_m]$

с известными результатами кластеризации действий $[(a_1, k_1), \ldots, (a_m, k_m)]$ кластер определяется следующим образом:

$$k_s = \arg\max_k \sum_{i=1}^{m} w_{a_i} \cdot [k = k_i], \quad \text{где } w_{a_i} \text{ обозначает вес действия } a_i$$

Аналогичная проблема с неравноценностью действий возникает и в подходе, использующем векторные представления сессий. На данный момент было рассмотрено два способа формирования векторного представления сессии:

1. усреднение векторных представлений действий.
2. использование One-Step-ALS для набора векторных представлений действий.

Первый подход может быть естественным образом обобщен на случай взвешенного усреднения:

$$e_s = \frac{1}{\sum_{i=1}^{m} w_{a_i}} \sum_{i=1}^{m} w_{a_i} \cdot e_{a_i}$$

Второй подход, One-Step-ALS, как уже было упомянуто в соответствующем разделе также поддерживает возможность учета важности векторов в наборе с помощью варьирования значений r_t, однако также можно рассмотреть следующий взвешенный функционал (для простоты здесь $r_t = 1$):

$$\mathscr{L}(s) = \sum_{i=1}^{m} w_{a_i}(\langle e_s, e_{a_i} \rangle - 1)^2 + \lambda \|e_s\|^2 \longrightarrow \min_{e_s}$$

Для него имеет место следующее аналитическое решение:

$$\hat{e}_s = \min_{e_s} \mathscr{L}(s) = (\sum_{i=1}^{m} w_{a_i} e_{a_i} e_{a_i}^T + \lambda \cdot I)^{-1} \cdot \sum_{i=1}^{m} w_{a_i} e_{a_i}$$

2.6 Кластеризация с использованием априорной информации

При осуществлении кластеризации в пространстве действий хочется использовать информацию о том, что действия из одной сессии относятся к одному кластеру. Этого можно достичь, например, при помощи агломеративной кластеризации [8].

В агломеративной кластеризации процесс группировки похожих объектов начинается с создания нескольких групп, где каждая из групп содержит один объект,

и затем последовательно две наиболее похожие группы объединяются в одну до тех пор пока не останется единая группа. Предлагается использовать следующую модификацию: вместо того чтобы начинать с групп которые содержат по одному объекту, можно начинать с групп, которые представляют собой действия из одной сессии. Таким образом все действия из одной сессии всегда будут принадлежать одному кластеру. Можно считать, что в данном случае производится кластеризация в пространстве сессий, но теперь каждая сессия представляется не одним вектором, а целым множеством векторов действий.

Рассмотрим классические способы подсчета расстояний между группами u и v в агломеративной кластеризации:

- single linkage:
$$d(u,v) = \min_{i,j}(dist(u[i], v[j])),$$
- complete linkage:
$$d(u,v) = \max_{i,j}(dist(u[i], v[j])),$$
- average linkage (UPGMA):
$$d(u,v) = \frac{1}{|u| \cdot |v|} \sum_{i,j} dist(u[i], v[j]),$$
- weighted linkage (WPGMA):
$$d(u,v) = \frac{1}{2}(d(s,v) + d(t,v)),$$
где кластер u был образован кластерами s и t.
- Ward's linkage:
$$d(u,v) = \sqrt{\frac{2|u||v|}{|u|+|v|}}\|c_u - c_v\|_2,$$
где c_s — центроид кластера s.
- centroid linkage (UPGMC):
$$d(u,v) = \|c_u - c_v\|_2,$$
- median linkage (WPGMC):
$$d(u,v) = \|w_u - w_v\|_2,$$
где w_s это итеративно обновляющаяся «медиана» кластера. Если кластер s образован кластерами s_1 и s_2, то $w_s = \frac{1}{2}(w_{s_1} + w_{s_2})$.

Заранее определить какой из спсобов окажется наилучшим в настоящей задаче затруднительно, это будет исследовано в разделе 3 в конкретных экспериментах. Можно отметить, что при использовании single linkage, все сессии, содержащие одинаковый трек, будут объединяться в одну группу.

3 Эксперименты

В предыдущем разделе было описано большое число различных реализаций представленной общей модели. Опишем конфигурации модели, которые будут сравниваться в последующих экспериментах:

1. Сессии из одного действия, формирование векторов путем усреднения.
2. Сессии из одного действия, формирование векторов при помощи One-Step-ALS.
3. Сессии по времени, формирование векторов путем усреднения, кластеризация векторов сессий.
4. Сессии по времени, формирование векторов путем усреднения, кластеризация взвешенных векторов сессий.
5. Сессии по времени, формирование векторов путем усреднения, кластеризация действий с последующим голосованием.
6. Сессии по времени, формирование векторов путем усреднения, кластеризация действий с последующим взвешенным голосованием.
7. Сессии по времени, формирование векторов путем усреднения, кластеризация действий при помощи агломеративной кластеризации (с подбором способа подсчета расстояния).
8. Сессии по времени, формирование векторов при помощи One-Step-ALS, кластеризация векторов сессий.
9. Сессии по времени, формирование векторов при помощи One-Step-ALS, кластеризация взвешенных векторов сессий.
10. Сессии по времени, формирование векторов при помощи One-Step-ALS, кластеризация действий с последующим голосованием.
11. Сессии по времени, формирование векторов при помощи One-Step-ALS, кластеризация действий с последующим взвешенным голосованием.

12. Сессии по времени, формирование векторов при помощи One-Step-ALS, кластеризация действий при помощи агломеративной кластеризации (с подбором способа подсчета расстояния).

3.1 Синтетические данные

3.2 Разметка детский/взрослый

4 Optional: Детекция совместных аккаунтов

5 Заключение

6 Список литературы

[1] *CAMRa '11: Proceedings of the 2nd Challenge on Context-Aware Movie Recommendation*, New York, NY, USA, 2011. Association for Computing Machinery.

[2] Payal Bajaj and Sumit Shekhar. Experience individualization on online tv platforms through persona-based account decomposition. In *Proceedings of the 24th ACM International Conference on Multimedia*, MM '16, page 252–256, New York, NY, USA, 2016. Association for Computing Machinery.

[3] Rui Chen, Qingyi Hua, Yan-Shuo Chang, Bo Wang, Lei Zhang, and Xiangjie Kong. A survey of collaborative filtering-based recommender systems: From traditional methods to hybrid methods based on social networks. *IEEE Access*, 6:64301–64320, 2018.

[4] Yifan Hu, Yehuda Koren, and Chris Volinsky. Collaborative filtering for implicit feedback datasets. In *2008 Eighth IEEE International Conference on Data Mining*, pages 263–272, 2008.

[5] Jyun-Yu Jiang, Cheng te Li, Yian Chen, and Wei Wang. Identifying users behind shared accounts in online streaming services. *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, 2018.

[6] Yehuda Koren and Robert Bell. *Advances in Collaborative Filtering*, pages 77–118. Springer US, Boston, MA, 2015.

[7] Muyang Ma, Pengjie Ren, Yujie Lin, Zhumin Chen, Jun Ma, and Maarten de Rijke. π-net: A parallel information-sharing network for shared-account cross-domain sequential recommendations. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR'19, page 685–694, New York, NY, USA, 2019. Association for Computing Machinery.

[8] Daniel Müllner. Modern hierarchical, agglomerative clustering algorithms, 2011.

[9] Yuanzhe Peng. A survey on modern recommendation system based on big data, 2022.

[10] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. Bpr: Bayesian personalized ranking from implicit feedback, 2012.

[11] Gábor Takács and Domonkos Tikk. Alternating least squares for personalized ranking. In *Proceedings of the Sixth ACM Conference on Recommender Systems*, RecSys '12, page 83–90, New York, NY, USA, 2012. Association for Computing Machinery.

[12] Koen Verstrepen and Bart Goethals. Top-n recommendation for shared accounts. In *Proceedings of the 9th ACM Conference on Recommender Systems*, RecSys '15, page 59–66, New York, NY, USA, 2015. Association for Computing Machinery.

[13] Shoujin Wang, Longbing Cao, Yan Wang, Quan Z. Sheng, Mehmet Orgun, and Defu Lian. A survey on session-based recommender systems, 2019.

[14] Zhijin Wang, Yan Yang, Liang He, and Junzhong Gu. User identification within a shared account: Improving ip-tv recommender performance. In *Symposium on Advances in Databases and Information Systems*, 2014.

[15] Markus Weimer, Alexandros Karatzoglou, and Alex Smola. Improving maximum margin matrix factorization. volume 72, 09 2008.

[16] Xinyu Wen, Zhaohui Peng, Shanshan Huang, Senzhang Wang, and Philip S. Yu. Miss: A multi-user identification network for shared-account session-aware

recommendation. In Christian S. Jensen, Ee-Peng Lim, De-Nian Yang, Wang-Chien Lee, Vincent S. Tseng, Vana Kalogeraki, Jen-Wei Huang, and Chih-Ya Shen, editors, *Database Systems for Advanced Applications*, pages 228–243, Cham, 2021. Springer International Publishing.

[17] Shuo Yang, Somdeb Sarkhel, Saayan Mitra, and Viswanathan Swaminathan. Personalized video recommendations for shared accounts. *2017 IEEE International Symposium on Multimedia (ISM)*, pages 256–259, 2017.

[18] Amy Zhang, Nadia Fawaz, Stratis Ioannidis, and Andrea Montanari. Guess who rated this movie: Identifying users through subspace clustering, 2014.

[19] Yafeng Zhao, Jian Cao, and Yudong Tan. Passenger prediction in shared accounts for flight service recommendation. In *IEEE Asia-Pacific Services Computing Conference*, 2016.